ELSEVIER

Signal Processing 41 (1995) 49–77

SIGNAL PROCESSING

Survey and bibliography of Arabic optical text recognition

Badr Al-Badr[a,*], Sabri A. Mahmoud[b]

[a]*Department of Computer Science and Engineering, Mail Stop FR-35, University of Washington, Seattle, WA 98195, USA*
[b]*Computer Engineering Department, College of Computers and Information Sciences, King Saud University, P.O. Box 51178, Riyadh 11543, Saudi Arabia*

Received 21 January 1994; revised 27 May 1994

Abstract

Research work on Arabic optical text recognition (AOTR), although lagging that of other languages, is becoming more intensive than before and commercial systems for AOTR are becoming available. This paper presents a comprehensive survey and bibliography of research on AOTR, by covering all the research publications on AOTR to which the authors had access. This paper introduces the general topic of optical character recognition (OCR), and highlights the characteristics of Arabic text. It also presents an historical review of the Arabic text recognition systems. Further, this paper reports on the state of the art in AOTR research, and lists the specifications of commercially available systems for AOTR. In this paper, we first underline the capabilities of different AOTR systems, and then introduce a five stage model for AOTR systems and classify research work according to this model. We devote a section to each of the stages of this model: preprocessing, segmentation, feature extraction, classification, and post-processing. In the preprocessing section, we emphasize handling degraded documents, and thinning of Arabic text. In the segmentation section, we discuss methods of segmenting Arabic text and categorize the methods into five general approaches. In the feature extraction and classification sections, we highlight the main techniques and analyze AOTR research works based on those techniques. We then discuss approaches for post-processing and show their relation to the Arabic language. We conclude by pointing problems and directions for future research on AOTR.

Zusammenfassung

Forschungsarbeiten über die optische Erkennung arabischer Texte (AOTR) werden mit zunehmender Intensität betrieben, obwohl sie gegenüber anderen Sprachen etwas verzögert sind. Kommerzielle Systeme für AOTR sind schon erhältlich. Diese Arbeit gibt eine umfassende Übersicht und Bibliographie zur Forschung über AOTR, die alle Forschungsveröffentlichungen einschließt, zu denen die Autoren Zugang hatten. Diese Arbeit führt in die allgemeine Thematik der optischen Zeichenerkennung (OCR) ein und hebt die Besonderheiten arabischer Texte hervor. Sie gibt außerdem einen geschichtlichen Überblick über Erkennungssysteme für arabische Texte. Zusätzlich wird über den letzten Stand der AOTR-Forschungen berichtet, und es werden die Spezifikationen kommerziell erhältlicher AOTR-Systeme angeführt. Zuerst unterstreichen wir die Möglichkeiten der verschiedenen AOTR-Systeme, danach stellen wir ein Fünf-Stufen-Modell für AOTR-Systeme vor und klassifizieren die Forschungsarbeiten anhand dieses Modells. Wir widmen jeder dieser Modellstufen einen Abschnitt: Vorverarbeitung, Segmentierung, Merkmalsextraktion, Klassifikation und Nachverarbeitung. Im Abschnitt über Vorverarbeitung heben wir die Behandlung beschädigter Dokumente

*Corresponding author. E-mail: badr@cs.washington.edu.

SSDI 0165-1684(94)00090-5

hervor und die Ausdünnung arabischer Texte. Danach diskutieren wir Methoden zur Segmentierung arabischer Texte and unterteilen die Methoden in fünf Ansätze. In den Abschnitten über Merkmalsextraktion und Klassifikation heben wir die wichtigsten Techniken hervor und analysieren die AOTR-Arbeiten in bezug auf diese Techniken. Danach diskutieren wir Ansätze für die Nachverarbeitung und zeigen ihre Beziehung zur arabischen Sprache. Wir schließen die Arbeit mit Hinweisen auf Probleme und auf zukünftige Forschungsarbeiten in AOTR.

Résumé

Le travail de recherche sur la Reconnaissance Optique de Textes Arabes (ROTA), bien que moins avancé que pour d'autres langues, devient plus intensif qu'avant, et des systèmes commerciaux de ROTA deviennent disponibles. Cet article présente un apercu et une bibliographie de la recherche sur le ROTA, couvrant toutes les publications sur le sujet auxquelles les auteurs ont eu accès. Cet article introduit le sujet plus géneral de la Reconnaissance Optique de Caractères (ROC), et met l'accent sur les caractéristiques du texte arabe. Il présente également un résumé historique des systèmes de reconnaissance des textes arabes. Plus loin, ce texte fait un "état des lieux" de la recherche sur la ROTA, et énumère les specifications des systèmes disponibles commercialement. Dans cet article, nous soulignons d'abord les capacités des differents systèmes de ROTA, puis introduisons un modèle à 5 niveaux pour ces systèmes, et classons le travail de recherche d'après ce modèle. Nous consacrons une section à chacun des étages de ce modèle: prétraitement, segmentation, extraction de caractéristiques, classification et post-traitement. Dans la section consacrée au pré-traitement, nous accentuons le traitement du texte arabe dégradé, et l'amincissement du même texte. Dans la section de segmentation, nous discutons les méthodes de segmentation des textes arabes et catégorisons les méthodes selon 5 approaches générales. Dans les sections d'extraction de caractéristiques et de classification, nous soulignons les techniques principales et analysons les travaux de ROTA basés sur les dites techniques. Nous discutons ensuite des approaches pour le post-traitement et montrons leurs relations avec la langue arabe. Nous concluerons en indiquant certains problèmes et certaines directions pour la recherche future en ROTA.

Keywords: Arabic optical text recognition; Optical character recognition; Template matching; Structural analysis; Structural, statistical, and neural classifiers; Segmentation; Cursive text recognition; Survey; Bibliography; Degraded text recognition

1. Introduction

Since the advent of writing as a form of communication, paper prevailed as the medium for writing. Electronic media has only recently started to replace paper. Because it conserves space and is fast to access, electronic media is constantly gaining popularity. The convenience of paper, its widespread use for communication and archiving, and the amount of information already on paper, press for quick and accurate methods to automatically read that information and convert it into electronic form.

The potential application areas of automatic reading machines are numerous. One of the earliest and most successful applications is sorting checks in banks, as the volume of checks that circulates daily has proven to be too enormous for manual entry [1]. Other applications include reading postal addresses off envelops and automatically sorting mail, helping the blind to read, reading customer-filled forms (tax forms, insurance claims, application forms), automating offices, archiving and retrieving text, and improving human–computer interfaces (e.g., pen-based computers) [2, 3].

Optical character recognition (OCR) is the branch of pattern recognition that studies automatic reading. The ultimate goal of OCR is to imitate the human ability to read at a much faster rate by associating symbolic identities with images of characters. A good typist can type 85 words per minute with an average of 3 mistakes per page. To match those figures, an OCR machine should recognize at least 99.9% of its input correctly [4] with all errors being rejections, and at a rate much faster than 5 characters per second. The practical importance of OCR applications, as well as the interesting nature of the OCR problem have led to great research interest and measurable advances in this field.

Although OCR predates computers, work on OCR intensified, both in industrial and research settings with the advent of computers in the mid-1940s [2]. The first commercial OCR machine came out in the early 1950s. The recognition logic of those early machines was hardware-based; they read mostly numbers in a specially stylized font. In the 1970s, machines with software recognition logic became available, reading text in specialized fonts like OCR A and OCR B [5]. The 1980s saw the advent of specialized, faster document readers, reading a wide range of typewritten fonts and pages that have multiple fonts. Now, commercial OCR systems for Latin characters are widely available on personal computers [6, 7]. Further, systems on the market can now read a variety of writing styles (e.g., handwritten, printed omni-font) and character sets including Chinese, Japanese, Korean, Cyrillic, and Arabic.

Since the mid-1940s, researchers have carried out extensive work and published many papers on character recognition. Most of the published work on OCR has been on Latin characters, with work on Japanese and Chinese characters emerging in the mid-1960s. Useful reviews and surveys of the field of OCR include the historical review of OCR methods and commercial systems by Mori [8], and the surveys of Govindan and Shivaprasad [2] and Mantas [3]. The survey of Impedovo et al. [9] focuses on commercial OCR systems, while the survey of Tian et al. [10] surveys the area of machine-printed OCR. Tappert et al. [11] and Wakahara et al. [12] survey on-line handwriting recognition. Nouboud and Plamondon [13] and Suen et al. [14] survey methods for recognition of handprinted characters, while Bozinovic and Srihari [15] and Simon [16] survey off-line cursive word recognition. Two bibliographies of the fields of OCR and document analysis are [17, 18]. Stallings [19] and Mori et al. [20] produced surveys on the recognition of Chinese machine- and handprinted characters, respectively.

Although almost a third of a billion people worldwide, in several different languages, use Arabic characters for writing (along side Arabic, Persian and Urdu are the most noted examples), Arabic character recognition has not been researched as thoroughly as Latin, Japanese, or Chinese. The first published work on Arabic opticaltext recognition (AOTR) may be traced back to 1975, and was by Nazif [21]. In his master's thesis he developed a system for recognizing printed Arabic characters based on extracting strokes, that he called radicals (20 radicals are used) and their positions. He used correlation between the templates of the radicals and the character image. A segmentation stage was included to segment the cursive text. Years later, Badi and Shimura [22] and Nouh [23] worked on printed Arabic characters, and Amin [24] on handwritten Arabic characters. In the 1980s, research work on AOTR increased considerably, a trend that is continuing in the 1990s. The only surveys on AOTR, of which the authors are aware, are by Amin [25], Shoukry [26], and Jambi [27]. Those surveys, however, have a limited number of references on AOTR.

This lag in research on Arabic text recognition, compared to other languages, may be attributed to (i) the lack of adequate support in terms of journals, books, conferences, and funding, and the lack of interaction between researchers in this field; (ii) the lack of general supporting utilities like Arabic text databases, dictionaries, programming tools, and supporting staff; (iii) the late start of Arabic text recognition; and (iv) the special characteristics of Arabic script. Research publications on AOTR regularly appear in the proceedings of the National Computer Conference of Saudi Arabia, the International Conference for Statistics, Computer Science, Social and Demographic Research of Egypt, and the International Conference on Multi-lingual Computing (Arabic and Roman Scripts); and in the following journals: the Arab Gulf Journal for Scientific Research, the Egyptian Computer Journal, and the Arabian Journal for Engineering and Sciences, in addition to the international journals and conferences.

While most commercial OCR systems accurately recognize only Latin, Japanese, and Chinese script, recently, systems that read Arabic have started to emerge. TextPert Arabic by CTA is advertised as an omnifont bilingual text reader that runs on Apple Macintosh computers with a peak recognition rate of 99% (sales brochure). MULTREC, which runs on IBM PC compatibles, was developed at the Institute of Oriental Studies, Saint

Petersburg, Russia. Its reported peak recognition rate is 99.5% [28]. IQRAA, which runs on IBM PC compatibles under windows, was developed by Arab Scientific Software and Engineering Technologies, Egypt. The system is based on neural networks and recognizes several fonts. It has a reported recognition rate of up to 99% on different fonts and sizes, without training, with a recognition speed of 1500 characters per min. We are not aware of any independent confirmation of these performance figures.

This paper surveys all the research publications on AOTR to which the authors had access. It also reviews general OCR techniques, with an emphasis on the issues pertinent to Arabic. All bibliographic references on AOTR that we have found are listed in the references section.[1]

The organization of the paper is as follows: Section 2 details the characteristics of Arabic text. The capabilities of AOTR systems is addressed in Section 3. Section 4 presents a general model for Arabic text recognition system, with subsections that address the five stages of the model (preprocessing, segmentation, feature extraction, classification, and post-processing). Future directions and conclusions are addressed in Sections 5 and 6, respectively.

2. Characteristics of Arabic text

The inadequate research activity on AOTR can be attributed in part to the special characteristic of Arabic text. The calligraphic nature of Arabic writing distinguishes it from other languages in several ways:

- Arabic text is written from right to left.
- Arabic has 28 basic characters, of which 16 have from one to three dots. Those dots differentiate between the otherwise similar characters. Additionally, three characters can have a zigzag like stroke (*Hamza* ء). The dots and *Hamza* are called secondaries and they are located above the character primary part as in *alif* (أ), below like *ba* (ب), or in the middle like *jeem* (ج) (see Table 1).

Table 1
The different shapes of Arabic characters

Character	Isolated	Beginning form	Middle form	End form
alif	ا	ا	ـا	ـا
ba	ب	بـ	ـبـ	ـب
ta	ت	تـ	ـتـ	ـت
tha	ث	ثـ	ـثـ	ـث
jeem	ج	جـ	ـجـ	ـج
hha	ح	حـ	ـحـ	ـح
kha	خ	خـ	ـخـ	ـخ
dal	د	د	ـد	ـد
thal	ذ	ذ	ـذ	ـذ
ra	ر	ر	ـر	ـر
zay	ز	ز	ـز	ـز
seen	س	سـ	ـسـ	ـس
sheen	ش	شـ	ـشـ	ـش
ssad	ص	صـ	ـصـ	ـص
dhad	ض	ضـ	ـضـ	ـض
tta	ط	طـ	ـطـ	ـط
ttha	ظ	ظـ	ـظـ	ـظ
ain	ع	عـ	ـعـ	ـع
ghain	غ	غـ	ـغـ	ـغ
fa	ف	فـ	ـفـ	ـف
qaf	ق	قـ	ـقـ	ـق
kaf	ك	كـ	ـكـ	ـك
lam	ل	لـ	ـلـ	ـل
meem	م	مـ	ـمـ	ـم
noon	ن	نـ	ـنـ	ـن
ha	ه	هـ	ـهـ	ـه
waw	و	و	ـو	ـو
ya	ي	يـ	ـيـ	ـي

- Written Arabic text is cursive when both machine-printed and handwritten. Within a word, some characters connect to the preceding and/or following characters, and some do not connect.

[1] For completeness, we have referenced papers that came out while this manuscript was being revised [29–33].

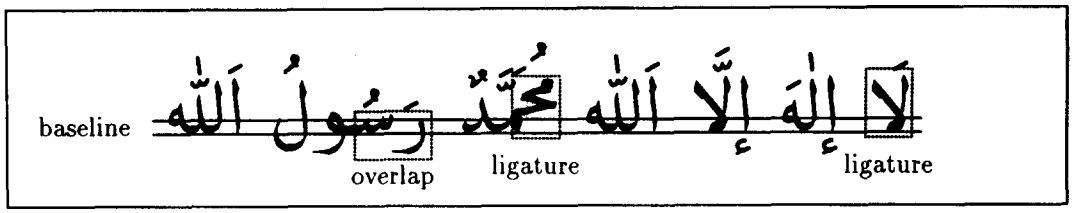

Fig. 1. A sample of written Arabic showing some of its characteristics.

The connectivities of characters result in a word having one or more connected components. We will refer to each connected piece of a word as a *sub-word*. As an example, the word *ketab* (كتاب) consists of two sub-words: *keta* (كتا), which consists of three characters, and *ba* (ب), which is a single character.

- The shape of an Arabic character depends on its position in the word; a character might have up to four different shapes depending on it being isolated, connected from the right (beginning form), connected from the left (end form), or connected from both sides (middle form). The different shapes of Arabic characters are displayed in Table 1.
- A distinguishing feature of Arabic writing is the presence of a baseline. The baseline is a horizontal line that runs through the connected portions of text (i.e., where the character's connection segments are located). The baseline has the maximum number of text (black) pixels (see Fig. 1).
- Characters in a word may overlap vertically (even without touching) (see Fig. 1).
- Arabic characters do not have a fixed size (height and width). Size varies across different characters and across the different shapes of the same character (see Table 1).
- Characters in a word can have diacritics (short vowels). These diacritics are written as strokes above the baseline such as *Fat-hah* (ـَ), *Dhammah* (ـُ), *Shaddah* (ـّ), *Maddah* (ـٓ), and *Sukun* (ـْ), or below the baseline such as *Kasrah* (ـِ). Moreover, *Tanween* may be formed by having double *Fat-hah* (ـً), *Dhammah* (ـٌ), or *Kasrah* (ـٍ). A different diacritic on a character may change the meaning of a word. Readers of Arabic are accustomed to reading un-diacritized text by deducing the meaning from context.
- Several characters can combine vertically to form a ligature, especially in typeset and handwritten text (see Fig. 1).

Fig. 1 demonstrates some of these characteristics n a typeset Arabic sentence consisting of seven words. Reading from right to left, the first word is a ligature made up of two characters, the second word consists of three characters and two connected components. The short strokes at the top of text are diacritic marks. The ligature in the middle of the figure consists of three vertically stacked characters.

Some of these characteristics greatly complicate recognition. One of the hardest problems with Arabic is its cursiveness; this is why segmentation is a crucial step for many AOTR systems. Many recognition errors are attributed to the segmentation phase, and a large portion of processing time is allocated to it. (This point is addressed in detail in Section 4.2.) Recognizing isolated Arabic characters is not fundamentally different from recognizing Latin text (aside from the larger number of classes of Arabic).

For Arabic, recognizing typeset text is generally harder than recognizing typewritten text because ligatures and character overlap (which characterize typesetting of Arabic text) complicate segmentation. For instance, in Fig. 1, if the word in the middle of the figure were typewritten, instead of typeset, the characters would have been written consecutively as 'محمد'. Furthermore, recognizing handwritten Arabic is even harder because of the variability in character shapes. Fig. 2 shows samples of handwritten, typewritten, typeset, and calligraphic Arabic text. In the next section we address the capabilities of different AOTR systems and the type of text they recognize.

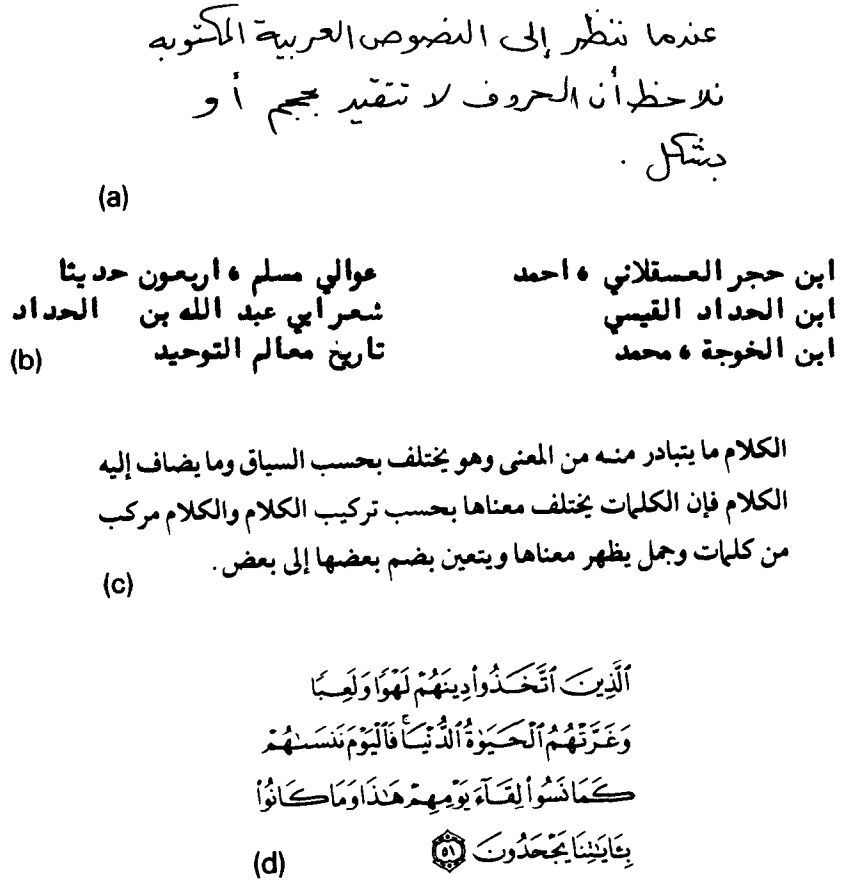

Fig. 2. Samples of Arabic text: (a) handwritten, (b) typewritten, (c) typeset, and (d) calligraphic.

3. Capabilities of OCR systems

Character recognition systems differ widely in how they acquire their input (on-line versus off-line), the mode of writing (handwritten versus machine-printed), the connectivity of text (isolated characters versus cursive words), and the restriction on the fonts (single font versus omnifont) they can recognize.

The first step in any recognition system is to capture text and transform it into digital form. Character recognition systems can capture their input text in two fundamentally different ways. *On-line* (or real-time) systems recognize text while the user is writing with an on-line writing device capturing the temporal or dynamic information of the writing. This information includes the number, duration, and order of each stroke (a stroke is the writing from pen down to pen up). The user typically writes on a digitizing tablet or a combination of screen/digitizer that instantaneously displays what the user writes. The writing here is represented as a one-dimensional, ordered vector of (x, y) points. Typical resolutions for digitizing tablets start at 200 points per inch while sampling rates are usually 100 points per second [11].

On-line systems are limited to recognizing handwritten text. Some systems recognize isolated characters [24, 34–41] and handwritten mathematical formulas [42, 43], while others recognize cursive words [22, 44–48]. Since the segmentation problem in Arabic is non-trivial the latter systems deal with a much harder problem.

Off-line systems, in contrast, recognize text that has been previously written or printed on a page

and then optically converted into a bit image. These systems are valuable since most of the text we are interested in recognizing is already in print. Here, a page of text is represented as a two-dimensional array of pixel values. Off-line systems do not have access to the time-dependent information captured in on-line systems [11]. Abuhaiba and Ahmed [49] describe a method to simulate that temporal information in off-line Arabic handwriting.

While some off-line systems use video cameras to digitize pages of text (e.g., [23, 50–55]), the trend now is to use scanners with resolutions ranging from 200 to 400 dots per inch (e.g., [56–62]). Scanners introduce less noise to an image, are less expensive, and more convenient to use for character recognition, especially when coupled with automatic document feeders, automatic binarization, and image enhancement.

Among the off-line systems that recognize handwritten isolated characters are [63–67]. Among the systems that recognize handwritten Arabic (Hindi) numerals are [50, 68, 69], and handwritten words are [44, 51, 70, 71]. The majority of off-line systems recognize typewritten cursive words [21, 52, 54, 56, 57, 60, 61, 72–80], while [23, 55,62, 81–84] recognize only typewritten isolated characters. The systems of [85–89] are designed to recognize typeset words. One of the systems [90], recognizes bilingual (Arabic/Latin) typewritten words. Examples of systems for recognition of other languages that use Arabic script are [67, 91, 92], which are designed for the recognition of Persian, Ottoman (Old Turkish), and Urdu, respectively.

Of the systems that recognize machine-printed text, some are hand-crafted for a certain font [83, 86, 93], some can be trained to recognize a font [57, 61, 72, 76, 82, 89, 90, 94, 95], while others can recognize different fonts at the same time [59, 77, 78, 80, 87, 88, 96]. In the following section we analyze the above research works by introducing a general model for AOTR and discussing the processing done at each stage of the model.

4. A general model for AOTR systems

The process of recognizing Arabic text can be broadly broken down into five stages: (1) preprocessing, (2) segmentation, (3) feature extraction, (4) classification, and (5) post-processing.

The *preprocessing* stage is a collection of operations that apply successive transformations on an image. It takes in a raw image and enhances it by reducing noise and distortion, and hence simplifies segmentation, feature extraction, and consequently recognition.

The *segmentation* stage takes in a page image and separates the different logical parts, like text from graphics, lines of a paragraph, and characters (or parts thereof) of a word.

The *feature extraction* stage analyzes a text segment and selects a set of features that can be used to uniquely identify the text segment. These features are extracted and passed in a form suitable for the recognition phase.

The *classification* stage is the main decision-making stage of an OCR system. The classification stage uses the features extracted in the previous stage to identify the text segment according to preset rules. This stage may use feature models obtained in an (off-line) training (modeling) phase to classify the test data.

The *post-processing* stage, which is the final stage, improves recognition by refining the decisions taken by the previous stage and recognizes words by using context. It is ultimately responsible for outputting the best solution and is often implemented as a set of techniques that rely on character frequencies, lexicons, and other context information.

Fig. 3 shows the different stages of AOTR systems. The preprocessing, segmentation, and post-processing stages are not necessarily executed by all Arabic optical text recognition systems. Some systems assume noise-free input. Also, some techniques recognize a character before segmentation, while others recognize isolated characters. Post-processing is addressed by only a few researchers of Arabic text recognition systems. In the next sections we address the different stages of the above model. Additionally, we briefly address learning.

4.1. Preprocessing

The recognition accuracy of OCR systems greatly depends on the quality of the input text and the

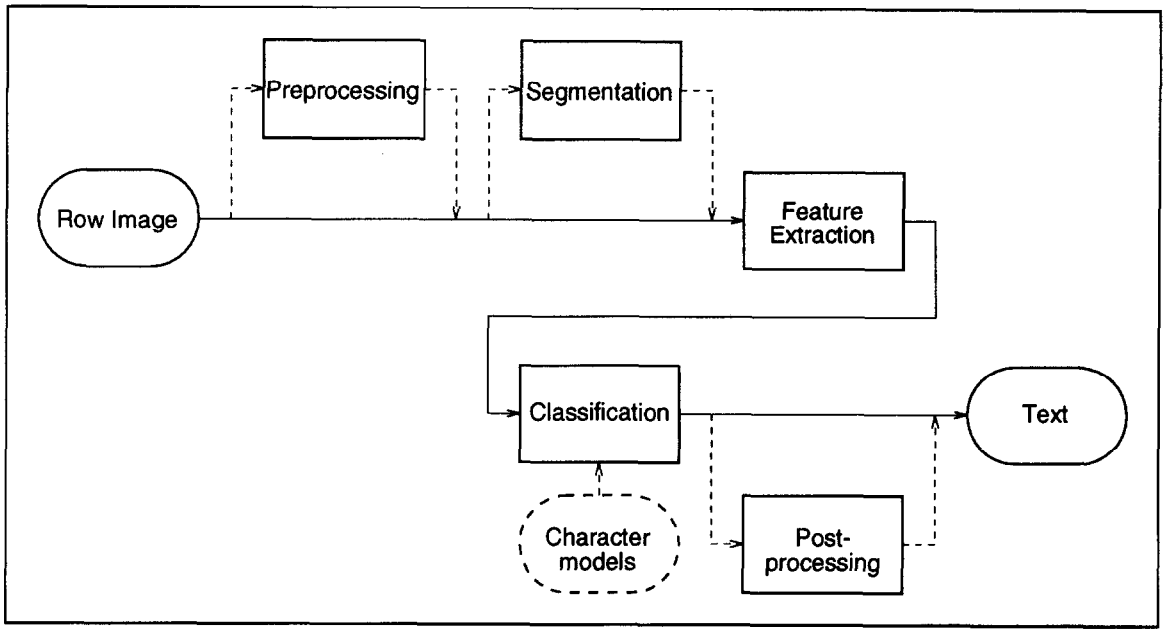

Fig. 3. A model for Arabic optical text recognition systems.

lack of noise, even more so than with humans. Baird [97] reports that even for OCR methods that perform well on some images, they perform much worse on images that are only slightly harder to human readers. The quality of input text depends on many factors.

Document history: A document that has been faxed or copied several times is harder to read than the original. Text gets thinner or thicker, salt and pepper noise appears, and contrast diminishes.

Printing process: A typeset document is clearer than a typewritten one, which in turn is clearer than the output of a dot-matrix printer. Other deformations that relate to the printing process include ink spreading, and ink chipping.

Font clarity: Exotic fonts, small font sizes, italic and bold characters, subscripts and superscripts, and using multiple font sizes (e.g., drop caps) and styles complicate recognition.

Paper quality: Opaque, heavy-weight, smooth, uniform grain paper is easier to read than light-weight, transparent paper (e.g., newspapers).

Document condition: The presence of extraneous markings and stains make reading harder.

Image acquisition: The digitization of on-line script is limited by tablet resolution and sampling rate, and often introduces distortions like small zigzags. The quality of scanned text is compromised by positioning variations (skew, translations, stretching, etc.), defocusing, unclean document glass, and the limited resolution.

Once text is acquired, either on-line or off-line, it should be preprocessed to simplify recognition. Preprocessing operations are usually specialized image processing operations that transform the image into another with reduced noise and variation. Those operations include binarization, filtering and smoothing, thinning, alignment, normalization, and baseline detection. Ideally, preprocessing should remove all variations and detail from a text image that are meaningless to the recognition method. As that goal is still illusive, preprocessing attempts to reduce noise and data variations as much as possible. Were it not for noise, distortions, and variations, recognition of isolated printed characters would have been solved long ago [98]. Preprocessing is especially important when recognizing handwriting to reduce the intrinsic variability

in writing styles. Brown and Ganapathy [99] postulate that a perfect preprocessing system would make the characters so uniform that recognition becomes trivial.

Little work has been done on characterizing noise that affects document images or on analyzing the effect of noise on recognition rate, especially for AOTR. In the literature, Baird [98] models optical distortions of document images (at the character level), like speckle, skew, and blur and calibrates that model [97]. He also surveys document degradation models in [100]. An alternate degradation model is suggested by Kunango et al. [101]. They present a global degradation model and model the distortion that occurs when photocopying thick bound documents (the black streaks and the reduction in size). Pavlidis [102] experimentally analyzes the effects of distortion on the recognition rate of a structural OCR system.

As two shades of gray suffice for text recognition, a gray scale image is converted to a bi-level image by the binarization (thresholding) process. An effective binarization method is to compute the histogram of the gray values of the image and then find a cut-off point (usually a valley between two peaks in the histograms). All gray values darker than the cut-off point are made one's (foreground) and the lighter values are made zero's (background). The binarization threshold should be carefully selected: If it is too high the text would grow thinner and connected pieces might break up, and if it is too low the text would grow thicker and isolated pieces might join. In many cases, documents do not have unique foreground and background colors, like when some of the text on the page is black on white background while the rest is white on black background. Here, binarization should be done on the uniform zones of the image and should invert images of zones that are on a black background. Lately, researchers have started investigating recognizing gray scale images, without prior binarization [103].

Conditioning steps like filtering and smoothing remove noise and other uninteresting variations in the image which are often byproducts of image acquisition. On-line systems track the movement of the stylus (pen) on the tablet. Some of the conditioning steps here are removing stray ending strokes, joining two strokes when they are sufficiently close to each other, and reducing vibration by eliminating points that make an angle of less than a prescribed threshold [24, 45, 46]. Another smoothing method for on-line text and skeletons and contours is to apply a low-pass filter to the vector of points representing the text [76]. An example of such a filter is one that averages each set of eight consecutive points to remove high frequency distortions [47]. Another example is using polygonal approximation [35].

A method that is often used to reduce noise in two dimensional (off-line) images is to traverse the image, pixel-by-pixel, using a 3×3 window and to change the value of a pixel based on the value of its 8-neighbors (north, northeast, east, etc.). That window, for example, can be used to fill in holes and eliminate isolated points [54, 75, 82, 95]. A one pixel is set to zero if there are not enough one pixels around it to support it and vice versa [91]. Mathematical morphology operations are often used for smoothing. The closing operation can eliminate small holes and fill gaps on the contours, while the opening operation can break narrow isthmuses and eliminate small islands, sharp peaks and capes [104]. For smoothing, the opening and closing operations are usually applied in succession [52] (see [105] for more details on this approach).

Thinning is an essential step for many OCR systems whose main task is reducing patterns to their skeletons. Pavlidis [106] defines the skeleton of a pattern as the set of points that has semi-equal distance from two or more points of the pattern contour. Thinning is often an efficient method for expressing structural relationships in characters as it reduces space and processing time by simplifying data structures. Examples of systems that use thinning/skeletonization are those of [36, 55, 81, 107]. Systems that recognize handwriting frequently use thinning to reduce the inherent variation in writing styles. Often, OCR systems traverse the skeleton of a word to simulate an on-line representation of it [49]. A typical thinning algorithm iteratively deletes points on the boundary of the pattern until it generates a one-pixel wide connected skeleton. It deletes a point on the boundary that is not an endpoint, not a break point, its deletion does not cause excessive erosion, and at least one of

its one 8-neighbors is not considered as deletable [108].

Many thinning algorithms are susceptible to noise in that the generated skeletons are sensitive to even small variations in the input pattern. Moreover, many of the algorithms deform junctions that should look like a 'T' and make them look more like a 'Y' and produce skeletons that have spurious tails that are not part of the input pattern [109]. For that, thinning is considered a major source of recognition errors in some systems [51, 72].

Another problem with many thinning algorithms that is particular to handwritten Arabic is that it reduces double and single dots (e.g., handwritten 'ـِـ' and 'ـِـ') to the same shape (a short line) [109]. That is the reason why dots are sometimes extracted from the image before thinning and are recognized separately. Some algorithms solve the problem by reducing a single dot to a single pixel instead of a short line at the expense of more computation time [55]. The interested reader can refer to [110] for an experimental comparison of the performance of 14 thinning algorithms, to [111] for an evaluation of the performance of ten thinning algorithms, and to [112] for a comprehensive survey of thinning methodologies.

On a more abstract level, the skeleton of an isolated handwritten character can be represented as a graph whose vertices are the endpoints, tree points (junction points), and edge points (points at which line slopes change) of the skeleton [63, 113]. Shaheen and Abo Samra [47] define the skeleton of a word by using vertical or horizontal line segments only; hence, they describe the pattern with a small set of primitives which they use as features.

Instead of iteratively deleting contour points, another class of thinning algorithms use clustering to directly produce a graph representation for a pattern's skeleton. Mahmoud et al. [109] use fuzzy clustering to partition the image of an isolated handwritten Arabic character into a set of c adjacent clusters, where c is a predetermined number of clusters and each cluster is a group of adjacent pixels of the image. The skeleton is then made up of line segments that pass through the centers of the clusters. The skeletons produced using this approach do not suffer from spurious tails nor deformed junctions, they are relatively stable under variations in the input pattern and a single dot is reduced to a pixel and not a line. Since this method is slow and can work only on isolated characters, the authors suggest the training stage as the best place to use this type of skeletonization. Abuhaiba et al. [114] develop an algorithm for estimating a suitable number of clusters for each character, for use with the algorithm of [109].

Since the sizes of Arabic characters greatly vary, size normalization is often used to scale characters to a fixed size and to center the character before recognition [50, 80, 81, 83]. This is useful in recognition methods that are sensitive to variations in size and position like template matching and correlation methods. Fakir and Sodeyama [89] measure the bounding box of a segmented printed character and apply (independent) expansion factors in the x and y dimensions to make the character fill a box of a certain size. To recognize different font sizes or to estimate the average width of a character for segmentation, researchers measure the average height of a character from the horizontal projection [53, 70].

The baseline in Arabic text contains valuable information about the orientation of the text and the location of connection points between characters. A common method for finding the baseline is using the horizontal projection. The horizontal projection histogram is simply a vector that has as many entries as there are rows in the image, with each entry containing the number of one pixels in its corresponding row. The vertical projection histogram is similarly defined for columns. The baseline shows up on the histogram as consecutive set of entries with maximum value (e.g., [75, 89, 91]). Some researchers estimate the baseline of all subwords on a line and line them up by shifting up or down [47]. Note that methods that use vertical and horizontal projections are effective only when the text is upright and its skew angle is minimal.

Many AOTR systems use baseline detection to align skewed pages, to separate the different lines of a text block, and to segment words into characters [60, 75, 77, 78, 115]. When scanning documents (especially when using an automatic document feeder) the resulting image can often be skewed. Fakir and Sodeyama [89] align (de-skew) a text image by using the Hough transform to estimate

the slope of the baseline and then use the slope to rotate the text image.

4.2. Segmentation

After the preprocessing stage, most OCR systems isolate the individual characters or strokes before recognizing them. Segmenting a page of text can be broken down into two levels: page decomposition and word segmentation. When working with pages that contain different object types like graphics, headings, mathematical formulas, and text blocks, page decomposition separates the different page elements, producing text blocks, lines, and sub-words. While page decomposition might identify sets of logical components of a page, word segmentation separates the characters of a sub-word. Generally, the performance of OCR systems as a whole crucially depends on how accurately they isolate the characters. As this statement is generally true for cursive text recognition, it is especially pertinent to Arabic, where characters connect within a word.

4.2.1. Page decomposition

Page decomposition is a sub-field of document analysis. Document analysis studies the structure of documents and the identification of the different logical parts in documents. Surveys of the field of document analysis are found in [18, 116, 117]. In works on Arabic that we have examined, page decomposition was limited to separating the different lines of a text block and extracting the sub-words. By far, the most common method of line separation is to use the horizontal projection histogram of the text image. The line-breaks in the text correspond to gaps (or minimas) in the histogram (e.g., [52, 60, 75, 118]). Note that some of those gaps might correspond to space between characters and their dots. One has to be careful not to separate dots above and below a line of text from their lines [61, 75]; for example, this method might fail when line spacing is tight and lower dots from one line occur on the same image row as the upper dots of the line beneath it. To deal with that, Parhami and Taraghi [91] and Khella [75] separate lines by first finding the baseline by horizontal projection and then grouping, as a line, all components that are closer to one baseline than any other.

Similarly, since an Arabic word might have one or more connected components, the most common method of sub-word separation is to use the vertical projection histogram of a line image and look for gaps or minimas [52, 60, 61, 75, 90, 94, 115, 118, 119]. Haj-Hassan [53] uses the vertical projection to directly separate words from each other and estimates, during training, the width of the minimum space between two words and conjectures that it is greater than any intra-word space. That width is then used to isolate the different words.

The histogram method, however, fails when characters in the different sub-words vertically overlap with each other. To handle that situation, other methods seek to identify the different sub-words and then shift them apart and insert a blank column between them. Sub-words can be identified by tracing their contours and detecting closed contours [73, 75, 77, 78], by tracing their skeletons [70], and by connected components labeling [91].

As an added step, some systems separate the dots and *Hamza* from the text, as well. This reduces the number of shape classes that the recognition stage should identify [75, 82, 95]. Al-Yousefi [58] uses this to prevent thinning from blurring the distinction between single and double dots. This separation is usually done by using horizontal projection [52, 53, 85, 89, 91, 96, 120] or by extracting the contours of these auxiliary characters [75, 82, 95].

4.2.2. Word segmentation

Segmentation methods for cursive [117, 121–124] and machine printed [125, 126] Latin text have been studied extensively [127]. Although some methods for cursive Latin might carry over to Arabic, in general they are insufficient for segmenting Arabic script. When reviewing methods that deal with connected characters in an Arabic word, we find five main approaches.

1. Assume that the input is already segmented into characters (i.e., do not address the problem of cursiveness).
2. Segment input words into primitives smaller than a character.
3. Segment input words into characters.

4. Recognize input words, with segmentation being a byproduct of recognition.
5. Recognize input words, as a whole, with no segmentation.

In the following we will discuss each of the five approaches.

Assume characters already segmented: Isolated Arabic characters are rarely used for writing (the main exception is in mathematical formulas [42, 43]). To be practical, systems that assume segmented input (e.g., [55, 63, 64, 81–83]) require a segmentation subsystem to segment the characters of a word before recognition.

Segmenting into primitives: This approach segments a cursive sub-word at all locations that appear to be connection points. This means segmenting the sub-word into primitives possibly smaller than a character, like strokes, intersection points, inflection points, and loops. The usual recognition scheme here is to recognize the primitives and then combine them into characters as exemplified in [35, 56, 77, 78, 94, 128]. This approach is mostly used to segment on-line and thinned off-line words. The line-image (skeleton) of a sub-word is traced and is segmented at locations that are likely to be connection points. In Arabic, the connection points between characters correspond to feature points that can be detected on the skeleton. Those feature points include: ends of lines, points at which lines cross, and points at which a line branches into two [51, 128], points at which a stroke rapidly changes slope [48], points at which local direction of the curves change [22, 44], cusp points (two strokes that meet at a small angle), and inflection points (points at which a curve changes direction) [35, 129].

Parhami and Taraghi [91] segment a typewritten sub-word into characters and sub-characters by identifying a series of potential connection points on the baseline at which line thickness changes from or to the thickness of the baseline. Although they also have some rules to keep characters at the end of a sub-word intact, they segment some of the wider characters (e.g., 'س') into up to three segments. Other systems segment a connected sub-word into primitives based on the vertical projection histogram where segmentation points are the locations where histogram falls below a certain threshold [56, 77, 78, 90]. However, these systems would not isolate a set of characters that are combined into a ligature [56, 94].

The advantage of segmenting into primitives and not characters is that it is easier to identify a set of potential connection points, which would include all the actual connection points, than to identify directly the actual points. It is then up to the classification stage to decide which are the actual connection points in light of the a priori information it has. This method is particularly appropriate for recognizing handwritten text where character boundaries are often ambiguous. However, when segmenting into strokes or primitives instead of characters, the recognition problem becomes that of recognizing the different strokes, primitives, or combined characters. This can significantly increase or reduce the number of shape classes to recognize depending on how a primitive is defined.

Segmenting into characters: This approach attempts to segment correctly a word into characters and then recognize the characters. In this approach, the segmentation step becomes the most critical step in the recognition process. Many of the techniques used here are similar to those used in the previous approach but are modified to prevent breaking up a character into more than one part. El-Sheikh and El-Taweel [130] segment an on-line sub-word by applying three sets of rules for the different positions in a sub-word: the first, the last, and the middle characters. Those rules relate to the average character width and relationships between the external points in the x- and y-axis. Such ad hoc rules, however, are not likely to generalize to different writing styles. El-Khaly and Sid-Ahmed [72] segment a thinned word into characters by following the baseline of the word and detecting when pixels start to go higher or lower than it. The IRAC II system [35, 45], for example, segments at cusp points. Since a character might have several cusps (e.g., three cusps in 'سـ') and since single-cusp characters have dots over or below them, the system examines the dots around a cusp to decide whether or not to segment.

An effective segmentation method is to search for connection points along the baseline. One of the most common methods of segmentation uses the

vertical projection histogram (of the whole sub-word text or the portion near baseline only) and defines the connection points to be the locations where the value of the histogram dips below a certain threshold [52, 53, 60, 80, 89, 96, 115]. Since those dips can occur within some characters, researchers use different precautions to prevent breaking up characters. For instance, some researchers use heuristic rules that prevent segmenting characters that are less than a certain width [59, 60, 89, 120]. Others modify the vertical projection histogram by multiplying each entry by the height of the column relative to the baseline. This has the effect of magnifying the distance from baseline so that points far from the baseline would not be considered as connection points [77, 78].

Other researchers use a variant of the vertical projection in which the sum in each column is not the total number of pixels but rather the distance between the extremal pixels in the vertical direction or extremal points on the contour [73, 77, 78]. A connection point, then, is a point on the baseline whose extremal distance is less than a threshold. Khella [75] combines the two techniques by using the vertical projection histogram and the difference between the heights of corresponding pixels on the contours to prevent segmenting characters that have holes (e.g., *ssad* ص) as they have some dip points. Also he uses special rules to prevent segmenting a character below the baseline or at the end of a sub-word. Some use the character width, estimated in the training phase, and the presence of dots to guarantee that a character is not segmented prematurely [60, 73]. However, estimating character widths from training data is difficult, especially for handwritten text. So, some researchers estimate the height of writing from the horizontal projection and assume a certain relation between the height of text and the average width of characters [53, 70, 80].

Some researchers use the contour of a sub-word for segmentation. Margner [76] segments a sub-word at locations where the (smoothed) upper contour changes curvature from positive (clockwise) to negative. Kurdy and Joukhadar [88] use the upper distance function of the sub-word, which is the set of the highest points in each column. They assign to each point of the function a token name by comparing the point's height to the height and token name of the point on its right. Using a grammar, they then parse the sequence of tokens of a sub-word to find the connection points.

The segmentation methods that use the vertical projection histogram are best suited for machine printed characters with stable widths (e.g., single font). Those methods are ineffective in segmenting handwritten or typeset text, which has ligatures. This is because the connection points are not along the baseline, and hence they cannot be detected on the vertical projection. Also with this method, special care should be taken when recognizing underlined and italic text.

Amin and Al-Sadoun [131] represent a chain-encoded (chain code is described in Section 4.4) word as a binary tree and traverse the tree to segment the word. First, they break up the word into segments delineated by junction points and then construct a binary tree for the word, where each node of the tree corresponds to a segment. A node includes the (smoothed) Freeman code for a segment, its length, and the number and position of dots, if any. They then segment the binary tree into characters by traversing it starting at the node that contains the first segment of the first character. They decide whether a node (segment) is the beginning of the next character or a continuation of the present character based on the length of the segment, its direction, and presence of loops.

In many systems that segment into characters, segmentation accounts for most recognition errors [87, 89], and takes the most processing time [46]. As an example, the IRAC system [24], which recognizes isolated handwritten characters, had a much better recognition performance than the IRAC II system, which had to segment the words [45]. To improve recognition performance some systems retry segmenting a sub-word into characters when they fail to recognize a previously segmented character [73, 87]. This recursive segmentation–recognition is particularly effective for reading handwritten and noisy text [121]; however, it is computationally expensive.

Recognition without prior segmentation: This approach recognizes the characters of a connected word in place (i.e., without segmenting in advance). Some of the systems that adopt this approach start

at the extreme right (beginning) of a sub-word and examine a set of columns (as wide as the narrowest character) and try to recognize the set as a character [44, 57, 61]. When that fails, they iteratively add columns to the set until they recognize it as a character. Once a character is recognized, it is removed from the sub-word and the process is repeated. Abdelazim et al. [57] report that most characters can be recognized by observing only part of the character (40% of it) which improves recognition speed.

One problem with this approach is that if the system fails to recognize a character in any part of the sub-word, especially near the beginning, the rest of the word would be unprocessed. To remedy this, researchers [61] use a backward recognition process that recognizes the sub-word left-to-right and is triggered if the system fails to recognize a middle character. As recognized portions of a sub-word must be removed before proceeding with the rest of the sub-word, they use a special cutting edge to remove each portion. This edge is simply a vertical line for non-overlapping characters and it depends on character shapes for overlapping characters.

Al-Badr and Haralick [86, 132] avoid the segmentation problem altogether by applying mathematical morphology operations on the whole page to find the locations where shape primitives are present. They then combine those primitives into characters and printout the character identities and their locations on the page.

Recognize input words as a whole with no segmentation: Systems that recognize a word as a unit usually use global matching techniques to match input words to words in a database, e.g., [46]. This approach, however, is limited to recognizing a set of predefined words (e.g., computer commands in pen-based computers) and is less useful for recognizing general text.

4.3. Feature extraction

Once an OCR system has an isolated pattern (character or primitive), its next step is to extract the features of the pattern and pass them along to the classifier to classify it. Feature extraction is one of the most difficult and important problems of pattern recognition [9]. The selected set of features should be a small set whose values efficiently discriminate between patterns of different classes, but are similar for patterns within the same class. The feature extraction step is closely related to classification because the type of features extracted here must match what the classifier expects.

The two main control approaches for feature extraction and classification are interleaved control versus one-step control. In interleaved control, an OCR system alternates between feature extraction and classification. In one such realization, the OCR system extracts a set of features from a pattern, and based on the feature values, classifies the pattern into a (small) number of categories. The system then extracts another set of features that are specific to each category and classifies the pattern [56, 66, 87, 88, 128]. In one-step control, the OCR system extracts all the required features from a pattern and then classifies the pattern [61, 64, 73, 82, 95, 119].

Among the different design issues involved in building an OCR system, perhaps the most consequential one is the selection of the type and set of features. Although the literature abounds with systems that use differing feature types, feature types can be categorized into four main groups: structural features, statistical features, global transformations, and template matching and correlation [2, 9]. In the following subsections, we address using these features for the recognition of Arabic text.

Structural features: Structural features describe a pattern in terms of its topology and geometry by giving its global and local properties. Structural features can highly tolerate distortions and variations in writing styles but extracting them from images is not always easy. The structural features used in the literature depend on the kind of pattern to be classified.

In the case of characters, the features include strokes and bays in various directions, end points, intersection points, loops, dots, and zigzags. Some researchers use the height, width, number of crossing points, and the category of the pattern (character body, dot, etc.) [35, 36, 44, 133]; the presence and number of dots and their position with respect to the baseline [39, 62, 82, 87, 95]; the number of concavities in the four major directions, the number

of holes, and the state of several key pixels [88]; the number of strokes or radicals and the size of the stroke's frame [24]; and the connectivity of the character [21, 45].

The features extracted from component curves, strokes, and contour segments include the direction of curvature (e.g., clockwise), the type of the feature point at which a curve was segmented (cross point, etc.) [22, 44]; the direction, slope, and length of strokes [48]; the length of a contour segment, the distance between the start and end point of the contour projected on the x- and y-axis, and the difference in curvature between the start and end points [76]; and the length of vectors in the four major directions that approximate a curve [47].

Sometimes the pattern is divided into several zones and several types of geometric features in each zone are registered and counted, with some features constrained to specific zones. Those features include the number of concavities, holes, cross points, loops, and dots; the number and length of the contour segments; the zone with maximum (minimum) number of pixels; and concavities in the four major directions [70, 75, 82, 91, 95].

Statistical features: Statistical features describe a pattern in terms of a set of characteristic measurements extracted from the pattern. In an appropriately designed feature space, all patterns of the same class map to a unique partition of the space [61].

The statistical features used for Arabic text recognition include: zoning, characteristic loci, crossings, and moments. Ali [34] uses zoning of pixels as features by dividing a character into overlapping or non-overlapping regions and using the densities of pixels in these regions as features. The characteristic loci method counts the number of zero segments and one segments a vertical line crosses in the pattern and the length of each segment [57, 84]. The crossings method counts the number of times a set of radial lines at different angles (e.g., 16 lines at 0°, 22.5°, 45°, etc.) cross the pattern [65]. This method tolerates distortions and small variations and is fast to calculate [9]. The moments method is one of the most popular statistical approaches used for pattern recognition. Abstractly, when a pattern is visualized as an nth degree polynomial, the nth moments are the coefficients of that polynomial. The moments of a pattern about its center of gravity are invariant to translation and can be normalized to be invariant to rotation and scale [58, 61, 72]. Some researchers, however, attribute the limitations of their system to the heavy computation involved in the moments approach and its sensitivity to the smallest variation in input [61].

In general, statistical features can be easily and quickly extracted from a text image; they can tolerate moderate noise and variation, and systems may be trained automatically for new fonts. The main difficulty, though, is to find an optimal set of features that would properly partition the space and are efficient to extract and classify.

Global transformations: The transformation scheme converts the pixel representation of the pattern to an alternate more abstract representation which reduces the dimensionality of features. One of the simplest transformations is representing the skeleton or contour of a pattern as a chain of direction codes. Direction codes can correspond to the eight major directions as in Freeman code [40, 41, 44, 82, 120], to six major directions in the case of hexagonal sampling [75, 95], or to unequal angle increments [51]. The resulting chain-code is often simplified to reduce redundancy and remove abrupt directional changes. Amin et al. [52, 59, 118] use a transformation based on the vertical and horizontal projections of a pattern. They represent the pattern as a string of primitives that indicate whether the length of the rows (columns) in the horizontal (vertical) projection is zero, greater than the average length for that projection, or less than the average. They then use several heuristic rules to collapse those strings.

Fakir and Sodeyama [89] use the Hough transform to represent the skeleton of a character as a set of line segments and then use the length, location, and slope of the line segments as features. Several researchers use Fourier descriptors derived from the contour points of a segmented character [62, 73, 75, 82, 95, 134]. Those descriptors are invariant to translation, rotation, and scaling and can tolerate moderate boundary variations. In general, transformation schemes can be easily applied and tolerate noise and variation. However, they sometimes require the use of additional features, in conjunction, to obtain high recognition rates [62, 75,

82, 95]. Global transformation features have the advantage of automating the training process for new fonts.

Template matching and correlation: Template matching and correlation methods basically compare a pattern pixel-by-pixel to a set of pattern templates; the pattern is considered to belong to the class of the template to which it is most similar. The interested reader can refer to [135] for a study of different feature extraction algorithms and correlation methods.

While this method is one of the fastest to execute, it is sensitive to distortion. For that, researchers use wide variations of this method. One basic variation is to choose several pixels in key locations for correlation [56, 90, 136]. Nouh et al. [137] describe an algorithm to select a set of connected pixels that is common to a number of characters, by matching each character with all other characters in every shift position. Those fragments are considered to be features or primitives that are used to match characters. Nazif [21] uses template matching between the templates of the radicals and the character image. Tolba et al. [138] match the histograms of the input characters to those of the templates, and El Ramly and El-Hamalaway [115] use correlation when the structural technique stage fails.

4.4. Classification

Classification in an OCR system is the main decision making stage in which the features extracted from a pattern are compared to those of the model set. Based on the features, classification attempts to identify the pattern as a member of a certain class. When classifying a pattern, classification often produces a set of hypothesized solutions instead of generating a unique solution. The (subsequent) post-processing stage uses higher level information to select the correct solution.

Historically, classification followed two main paradigms: syntactic (or structural) and statistical (or decision theoretic) classification. Recently, recognition using neural networks has provided a third paradigm.

Syntactic methods: Under this paradigm, an input pattern is classified in terms of its components (pattern primitives) and the relations among them. The classifier first identifies the primitives of a character and then parses strings of primitives according to a given set of syntax rules [22, 42, 44, 53, 76, 128].

The most popular classification method here is to represent characters as production rules whose left-hand side represents character labels and whose right-hand side represents strings of primitives [67]. The right-hand side of rules are compared to the string of primitives extracted from a word. When there is a match, the primitive is considered an instance of the corresponding character. The matching process usually tries to match the largest number of components with the left-hand side of the rules [128]. In another form, a character is represented syntactically as a tree whose internal nodes are primitives and leaves are character labels. Classifying a character, hence, corresponds to finding a path through the tree to a leaf [56, 68, 75, 129]. To allow for missing and extraneous primitive, some systems attempt to recognize a string of primitives in the reverse direction when recognition in the forward direction fails [76]. Others use the fuzzy set theory of directions and model isolated handwritten characters as fuzzy attributed graphs; they then compare the fuzzy graph of the input character to those of the models [63].

Syntactic methods are especially popular for classifying handwritten text. The use of syntax to express structure is a disadvantage of the structural approach, since patterns can have infinite variations and do not always adhere to the strict mathematical constraints set by the theory of formal languages [2].

Statistical methods: Statistical classification consists of mapping fixed-length vectors of features into a partitioned space. Classification here can be as simple as a distance classifier. One form of the distance classifiers is to compare the features of a pattern with the mean value of features for each class and label the pattern with the label of the class to which it has closest feature values [76].

Instead of comparing to class means, the nearest-neighbor method compares an unknown pattern to a set of patterns that have been previously labeled with class identities in the training (modeling) stage. A pattern is identified to be of the class of the

pattern to which it has the closest distance [24, 34, 62, 66, 72, 75, 82, 95, 133]. The distance comparison is sometimes computed to K labeled patterns and not only one (K-nearest neighbor). El-Wakil and Shoukry [40] use a three-level classification scheme. While the first level is a dictionary lookup, the second is a 1-nearest-neighbor classifier, and the third is a K-nearest neighbor. One of the most critical issues in those methods is choosing an efficient and accurate distance or similarity measure. Some researchers use dynamic programming to calculate the cumulative distance between the features of the main body of a character to be recognized and the reference patterns [89].

Another common statistical method is to use Bayesian classification. A Bayesian classifier computes the a posteriori probability of each pattern class based on the detected features, the conditional probability of the features given a class, and the a priori probability of the class [56–58, 68, 134, 139].

Instead of examining all the features at once, decision tree classifiers arrange tests in the structure of a tree. Each node of the tree is a test on a feature and each outcome of the test leads to another node in the tree. The leaves of the tree are labeled with class identities. When a series of tests on a pattern lead to a leaf, the pattern is labeled with the label of the leaf [35, 48]. To improve accuracy, some systems use four decision trees, one for each connectivity form of a character (isolated, right-connected, etc.) [52, 59, 96].

The main advantage of statistical classifiers is that they can be automatically trained. The literature includes some simpler methods that do not fall under either paradigm, like dictionary lookup [41, 61], rule based classification [136], and handcrafted tree classifiers [37, 38].

Neural network classifiers: Artificial neural networks (called neural nets) are an emerging paradigm for pattern recognition. Neural networks can use interconnected networks or simple (and typically) non-linear units [140]. Artificial neural networks may be characterized according to network topology, characteristics of the artificial neurons, and the learning or training algorithm used. Generally speaking, the neural network approach to pattern recognition is non-algorithmic (or black box strategy) and is trainable. The black box is trained to learn the correct classification output for each of the training examples. The a priori knowledge and detailed knowledge of the internal system operation is minimal. In the training stage, the network self-organizes to achieve the required relationships between the inputs and outputs. For more details, reference may be made to [141–143].

The only publication that the authors are aware of on the subject of AOTR using neural networks is the paper of Ahmad [119]. However, the interest in using artificial neural networks for AOTR is mounting. A paper that is to appear on the subject is [144].

4.5. Hybrid approaches

To improve recognition performance, a trend now is to build hybrid systems, which use diverse feature types and combinations of classifiers arranged in layers. As a result, increasingly many researchers now use combinations of the above feature types and classification techniques.

Almuallim and Yamagochi [128] use three levels of feature extraction/classification to recognize handwritten words. Using the structural features of a stroke, they map it into one of five groups. They, then, compute a feature vector for the stroke and classify it by finding its distance to a set of identification vectors. Finally, they convert a string of identified strokes into characters by syntactic parsing.

Abdelazim and Hashish [56, 90] use a four-step approach. First, they use structural features for clustering character primitives into groups. Then they use a decision tree classifier that tests pixel locations in a primitive to produce a set of candidates. They then iteratively use correlation until they determine the exact identity of the primitive. Finally they reconstruct the characters from the recognized primitives.

Amin [96] segments a printed word into characters and divides each character into as many as seven primitives using the length of the rows (columns) of the horizontal and vertical projections as features. The complimentary characters are identified by using their density. The projection

histogram is coded as a string whose symbols are indications where a row (column) is of length zero, greater than the average for that projection, or less than the average. He then classifies the simplified strings using four decision trees, one for each form of an Arabic character.

Khella [75] uses a tree structure to group the Arabic character set based on the number and location of dots and holes. Then he uses a statistical classifier to identify the characters of the same group. Khella and Mahmoud use statistical classifier to identify the models and use the dots and holes to identify characters [82, 95].

Some systems do not follow the classical paradigm of pattern recognition which is feature extraction followed by classification. Al-Badr and Haralick use a set of 30 shape primitives as features. They detect those shape primitives on a page image using mathematical morphology operations. A character is defined as a set of primitives at a certain configuration relative to one another. Whenever the primitives of a character are found with the right configuration, the character is detected [86].

4.6. Learning

Learning here refers to two different phenomena: training and adaptation. Training (or modeling) is the process of teaching an OCR system the descriptions of characters using data labeled with character names. This process is mainly done off-line (i.e., before recognition). This is one of the main features of statistical classifiers, as the system designer does not need to build the classifier manually. Bayesian classifiers, for example, learn the conditional probability of the features given the characters [57], while nearest neighbor classifiers use the proximity to training data for classification [40].

Adaptation, on the other hand, is the process of improving the performance of an OCR system by benefiting from previous experiences. Some systems allow the user to identify a character when they fail to recognize it. They use the user input to recognize the character when they encounter it next time [129]. Adaptation is especially effective for adapting to a particular user's handwriting [70].

4.7. Post-processing

The final stage in the recognition process is post-processing. One of the objectives of post-processing is to improve word recognition rate (as opposed to character recognition rate). Post-processing is often implemented as a set of techniques that rely on character frequencies, lexicons, and other contextual information. As classification, sometimes, produces a set of possible solutions instead of a unique solution, post-processing is responsible for selecting the right solution using higher level information that is not available to the classifier. Post-processing also uses that higher level information to check the correctness of the solutions returned by the classifier. The most common post-processing operations are spell checking and correction. Spell checking can be as simple as looking up words in a lexicon. To improve the speed of lookup, a lexicon is sometimes represented as a tree [52, 96, 120]. When spell checking fails, Amin and Mari [120] use the Viterbi Algorithm to find alternate words whose characters, with a high probability, can be interchanged with the original ones, using a Hidden Markov model for each word.

Traditionally, when looking up Arabic words in a dictionary, one has to first remove prefixes and suffixes and then reduce the resulting word to its root. This method of representing a word as a root and a pattern (the pattern specifies how the root relates to the original word) can considerably reduce the size of the dictionary (at the expense of some computation). To save time, Abdelazim and Abdel-Mageed [133] check the spelling of recognized words by removing prefixes and suffixes and then looking them up a lexicon of 30 000 words and then verifying suffixes and prefixes. If a word is not in the lexicon, a spelling aid suggests a number of alternatives based on editing distance, phonetic rules, shape similarity, and the classifier's confidence in the recognized characters. They report that spell checking raised the recognition rate from 92% to 99%. A more detailed paper on this method of spell checking and correction is [145].

Amin and Al-Fedaghi [59] describe an elegant method for spell correction of Arabic words. They correct spelling errors and complete words that have some unrecognized characters using an

algorithm that depends on the frequencies of roots and patterns in Arabic. Post-processing here was effective in increasing word recognition rate by 7%. In an earlier work, Amin [45] constructed all possible words from the classifier's solutions. He then used the rules of phonetics, along with a small lexicon to rule out incorrect words. In another work, Amin et al. [129] syntactically and semantically analyzed words in a sentence to choose among a set of proposed words as a recognition for a sentence. The analysis involved verifying the compatibility of the subject and the object in accordance with the action implied by the verb. They then translated this sentence to French and used a speech synthesizer to convert text to speech. El-Sheikh [42] uses a context-free grammar to obtain precedence relations and, hence, parse a detected handwritten mathematical formula.

5. Future directions

Along with speed, the most important requirement of an OCR system is to recognize its input correctly. As stated earlier, to match the performance of an average typist, the minimum acceptable recognition rate is 99.9%, and speed is five characters per second, with most errors being rejections rather than substitutions. Practical experience suggests that a substantial portion of the total recognition cost is allocated to editing manually the mistakes of the OCR system [146].

Latin OCR systems can achieve rates of up to 99.9% on machine-printed, high quality, homogeneous font documents [4], and can top 99% for on-line recognition [11, 13]. Currently, AOTR is far from reaching the 99.9% goal. Table 2 summarizes some recognition techniques and rates of some AOTR systems. Note that since most of the performance results reported in the literature are not statistically valid, we cannot use them to compare the performance of OCR systems. Most of the performance results reported either in the literature or by manufacturers of OCR systems are derived from test sets that are small, font-specific, noise-free, or might have been used in the design, with little (if any) analysis of the results. In light of that, the results reported in the papers should be interpreted, generally, as the best possible performance, under the most favorable conditions. Analyzing experimental results can determine how the different parts of an OCR system account for recognition errors. A good example of error analysis is in [59] which reported that the most common error is single character rejection and substitution.

The field of AOTR crucially needs a standard set of test documents, in both image and character formats, and a set of performance evaluation tools. This would truly enable comparing the performance of different AOTR systems. Several standard test sets are available for English in the form of CD-ROM. The United States Postal Research office and the National Institute of Standards and Technology produced disks that store samples of handwritten data. The University of Nevada, Las Vegas, Information Science (UNLV) Research Institute produced a large volume of scanned documents. The IEEE and ACM distribute scanned versions of their journals and conference proceedings [146]. The Intelligent Systems Laboratory at the University of Washington has recently released a CD-ROM that includes about 1000 scanned pages (and their associated text files) from technical documents in different degrees of degradation, as well as document degradation software and performance evaluation software [147].

Most AOTR systems do not directly address the issue of degraded and distorted input. As stated earlier, in daily life it is hard to find documents in perfect condition. The use of noise models can greatly improve OCR performance for distorted input. Having a distortion model for the input document allows studying the degradation in recognition rate as a function of the distortion in the input document. Under such a model, distortion in the input can be varied in a controlled and continuous manner. Moreover, characterizing the degradation process facilitates designing effective preprocessing algorithms to reverse the degradation [101].

Currently, the main research direction in AOTR appears to be in producing systems that have better recognition rates. As the major difference between Arabic and Latin recognition is the need for a segmentation stage, a major research focus is in finding better segmentation algorithms (e.g., [131]) or

Table 2
The characteristics and performance of some Arabic OCR systems

References	Capabilities	Segmentation	Features	Classification	Post-processing	Performance
Abbas [50]	Handwritten numerals		n-tuple	Statistical		CR 85%, 5% reject
Abdelazim [90]	Machine printed bilingual text	Primitives	Correlation, selected pixels	Clustering, decision trees, reconstruction of characters	Lexicon	CR 95–99% at 55–90 WPM
Abdelazim [68]	Handwritten numerals		No. of crossings	Structural, decision trees		CR 98%
Abdelazim [56]	Machine printed text	Primitives	Structural, statistical	Structural, decision trees, statistical		CR 99% CR 99%
Abdelazim [57]	Machine printed text	After recognition	Characteristic loci	Bayesian classification		CR 98%
Abdelazim [85]	Typeset text	Primitives	Correlation	Clustering, nearest neighbor, reconstruction		CR 96% at 30 WPM
Abdelazim [133]	Machine printed text	Primitives	Structural, statistical	Nearest neighbor	Spelling correction	CR 99%
Ahmad [119]	Machine printed text	User assisted		Neural network		CR 80–90%
Al-Emami [48]	On-line	Primitives	Structural, chain-code	Decision tree		WR 86–100%
Ali [34]	Handwritten numerals	Thinning	Zoning of pixels	Table lookup		CR 98%
Almuallim [128]	Handwritten words	Primitives	Structural	Syntactic, distance		CR 91%
Al-Yousefi [64]	Handwritten characters		Moments	Bayesian classifier		CR 98.8%
Amin [24]	On-line characters		Chain-code, structural	Nearest neighbor		CR 95.4%
Amin [45]	On-line words	Characters	Structural	Table lookup	Phonetics, lexicon	CR 80%
Amin (IRAC III) [46]	On-line words	Whole word classification	Structural	Table lookup		CR 90%
Amin [129]	On-line words	Characters	Structural	Syntactic	Syntactic & semantic analysis, translation into French	
Amin [96]	Multifont text	Characters	Histogram features	Decision trees	Lexicon	CR 90%
Amin [120]	Multifont text	Characters	Chain code	Decision trees	Lexicon, Viterbi algorithm	CR 90% at 180 CPM
Amin [59]	Multifont text	Characters	Histogram features	Decision trees	Spelling correction, word completion	CR 95.5% (WR 88.51%) 180 CPM

Badi [22, 44]	On-line words	Primitives	Structural	Handcrafted tree, syntactic		CR 90%
Eldabi [61, 183]	Machine-printed text	After recognition	Moments	Table lookup		CR 94% at 10.8 CPM
El-Desouky [66]	Handwritten characters		Chain code	Clustering, matching distance		CR 94% at 180 CPM
El Gowely [87]	Multifont typeset text	Characters (recursive)	Statistic	Decision tree, heuristic rules		CR 94% at 240 WPM
El-Khaly [72]	Machine-printed words	Characters	Moments	Distance		CR 95–100%
El-Sheikh [73]	Machine-printed text	Characters (recursive)	Fourier descriptors	Pairwise classifier		CR 99%
El-Sheikh [37, 130]	On-line characters		Structural	Handcrafted tree		CR 99.6%
El-Sheikh [42]	On-line math formulas		Syntactic	Handcrafted tree	Syntactic parsing	CR 99%
El-Wakil [39]	On-line isolated characters		Chain code	Matching table/ nearest neighbor		CR 93%
El-Wakil [40]	On-line characters		Chain code, Structural	Matching, distance *k*-nearest neighbor		CR 93% at 240 CPM
Fakir [89]	Handwritten and typeset text	Characters	Hugh transform	Dynamic programming, distance		CR 80% hand, 97% machine
Goraine [51, 74]	Handwritten and typeset words	Primitives	Chain code, structural	Table lookup, reconstruction	Lexicon	CR 90% hand, 92% machine
Haj-Hassan [80]	Machine-printed text	Characters	Structural	Syntactical		CR 99% at 18 CPS
Jambi [70, 173]	Handwritten words	Characters	Structural	Table lookup		CR 79%
Kurdy [88]	Multifont typeset text	Characters	Structural, selected pixels	Multilevel, table lookup, heuristic		CR 98% at 1.33 PPM
Margner [76]	Machine-printed text	Primitives	Structural	Nearest neighbor		CR 96.9–99%
Nurul-Ula [83, 136]	Machine-printed characters		Selected pixels	Rule based		CR 100%
Parhami [91]	Typeset Farsi text	Primitives	Structural	Table lookup		CR 100% (large fonts)
Saadallah [41]	Handwritten characters		Chain code	Table lookup		CR 80–90%

CR: Character recognition rate; WR: Word recognition rate; WPM: Words per minute; CPM: Characters per minute; PPM: Page per minute.

using recognition methods that do not require segmentation (e.g., [86]). The paradigm of separate segmentation and recognition stages has an obvious weakness in that mistakes in segmentation are fetal, since they lead to mis-recognition. Researchers are now using iterative segmentation and recognition stages. One of the main variants of this paradigm is that when the recognition of a segmented pattern fails, the OCR system retries to resegment the pattern [73, 87]. Another variant is for segmentation to suggest a set of connection points, and to leave it up to the recognition phase to decide which points are the actual connection points [121].

Since each of the traditional feature types and classification methods has its strengths and weaknesses, we have described hybrid systems that combine several features types and classification methods to improve performance. Another way to improve performance is to use multiple classifiers simultaneously and then combine the ranked results produced by the classifiers [10, 146]. For a survey of methods of combining multiple classifiers, see [148]. For more effective post-processing, OCR systems can use even higher level information, like information on the subject of the document, language semantics, and information on the uniformity of type font, style, and size within words and paragraphs. With the technology of AOTR advancing, it might be time to study recognizing text with diacritic marks, and calligraphic text, which is even harder to recognize than typeset documents. A classic example of such writing is the text of the Holy Qura'an (Fig. 2(d)).

As OCR systems become more advanced, researchers are putting less emphasis on the character level and more emphasis on the document level. One such effort is in understanding page layout. This would allow automatically finding the correct reading order for multi-column and multi-lingual documents and separating the text from illustrations, mathematical formulas, and figures. Layout analysis should allow recognizing multi-lingual documents by automatically separating and identifying the language of the homogeneous zones on the page. Finally, researchers are now starting to look at methods for total document recognition where the format of the document is recognized along with its text. This would enable reconstructing the format of the input document which is often desirable [146].

6. Conclusion

This paper presented a comprehensive survey of the research on Arabic optical text recognition (AOTR) from the first publication, in 1975, to date. Research on AOTR is considerably lagging that of other languages (e.g., Latin, Chinese, and Japanese). This lag may be attributed to the late start (mid-1970s for Arabic compared with the mid-1940s for Latin), the special characteristics of Arabic text, which is cursive in both machine printed and handwritten, and the lack of support in terms of funding, manpower, programming utilities, text databases and dictionaries, and hardware. However, recent years have shown a considerable increase in the number of researchers and publications addressing AOTR. The last few years have also witnessed the emergence of several commercial systems for Arabic OCR (although not as sophisticated as those for Latin).

In this paper, we have detailed the characteristics of Arabic text that influence recognition. We have introduced a general model for AOTR systems. We then addressed each of the five stages of the model (preprocessing, segmentation, feature extraction, classification, and post-processing). At each stage, we analyzed and categorized AOTR research methods, and indicated limitations, peculiarities and advantages of those methods. Segmentation can be done at the page level and the word level. At the word level, we have identified five main approaches to segmentation. The main feature types used in AOTR include structural, statistical, global, and template matching and correlation, while the main classification approaches include syntactic, statistical, and neural approaches. Additionally, we have investigated systems that use hybrid feature types and classification approaches. We then discussed the insufficient use of post-processing in works on AOTR, which is mainly due to the lack of Arabic computerized dictionaries.

Our investigation indicates the need for more intensive research on Arabic text recognition and its supporting fields. The field of AOTR, crucially requires databases of Arabic document images (and their associated text) that have enough data for researchers to assess and report the performance of their work. Further, the analysis of the effects of image degradations in AOTR is limited, and more work is needed in this area. Perhaps the greatest difference between processing Latin and Arabic text is the need for segmentation. One of the biggest obstacles in the effort to improve AOTR is the lack of effective and efficient solutions to the segmentation problem. Research on segmentation, in particular, and recognition, in general, should focus on treating degraded text, italic and skewed text, text with overlapping characters and ligatures, text with diacritics, and omnifont and calligraphic text.

To improve recognition rates, researchers can use hybrid systems, systems with multiple classifiers, and lexicons and contextual information. To improve speed, it might be necessary to use parallel processing and to implement some OCR functions in hardware. As the use of bilingual and multilingual documents is increasing, systems for recognizing such text are necessary.

Finally, and most importantly, researchers of this field must work to increase communication and cooperation among them. Researchers of this field should set plans to tackle the above stated problems and limitations collectively, in light of the lack of support from companies and government agencies.

Acknowledgments

The authors acknowledge the comments and advice of Dr. Robert Haralick, which improved the technical content of the paper, and the help of King Abdulaziz City for Science and Technology in acquiring many of the listed references. The authors would also like to express their gratitude to the referees for their constructive criticism. The incorporation of their suggestions and comments improved the clarity of the final manuscript.

References

[1] J.R. Ullmann, "Applications of character recognition", in: K.S. Fu, ed., *Applications of Pattern Recognition*, CRC Press, Boca Raton, FL, 1982, Chapter 9.

[2] V.K. Govindan and A.P. Shivaprasad, "Character recognition – A review", *Pattern Recognition*, Vol. 23, No. 7, 1990, pp. 671–683.

[3] J. Mantas, "An overview of character recognition methodologies", *Pattern Recognition*, Vol. 19, No. 6, 1986, pp. 425–430.

[4] S. Kahan, T. Pavlidis and H.S. Baird, "On the recognition of printed characters of any font and size", *Pattern Recognition*, Vol. 9, No. 2, 1987, pp. 274–287.

[5] J.W. Smith and Z. Merali, *Optical character recognition*, The British Library, Wetherby, West Yorkshire LS23 7BQ, UK, 1985.

[6] D. McClelland, "OCR: Teaching your Mac to read", *Macworld*, November 1991, pp. 169–178.

[7] E.M. Welch, "Can you read this? OCR software", *MacUser*, Vol. 9, No. 8, November 1993, pp. 169–178.

[8] S. Mori, C.Y. Suen and K. Yamamoto, "Historical review of OCR research and development", *Proc. IEEE*, Vol. 80, No. 7, July 1992, pp. 1029–1057.

[9] S. Impedovo, L. Ottaviano and S. Occhinegro, "Optical character recognition – A survey", *Internat. J. Pattern Recognition and Artificial Intelligence*, Vol. 5, No. 1, 1991, pp. 1–24.

[10] Q. Tian, P. Zhang, T. Alexander and Y. Kim, "Survey: Omnifont printed character recognition", *Visual Communications and Image Processing '91: Image Processing*, Vol. SPIE 1606, Boston, MA, 1991, pp. 260–268.

[11] C.C. Tappert, C.Y. Sen and T. Wakahara, "The state of the art in on-line handwriting recognition", *IEEE Trans. Pattern Anal. Machine Intell.*, Vol. 12, No. 8, August 1990, pp. 787–808.

[12] T. Wakahara, H. Murase and K. Odaka, "On-line handwriting recognition", *Proc. IEEE*, Vol. 80, No. 7, July 1992, pp. 1181–1194.

[13] F. Nouboud and R. Plamondon, "On-line recognition of handprinted characters: Survey and beta tests", *Pattern Recognition*, Vol. 23, No. 9, 1990, pp. 1031–1044.

[14] C.Y. Suen, M. Berthod and S. Mori, "Automatic recognition of handprinted characters – The state of the art", *Proc. IEEE*, Vol. 68, No. 4, April 1980, pp. 469–487.

[15] R. Bozinovic and S. Srihari, "Off-line cursive script word recognition", *IEEE Trans. Pattern Anal. Machine Intell.*, Vol. 11, No. 1, January 1989, pp. 68–83.

[16] J.-C. Simon, "Off-line cursive word recognition", *Proc. IEEE*, Vol. 80, No. 7, July 1992, pp. 1150–1161.

[17] F. Jenkins and J. Kanai, A keyword-indexed bibliography of character recognition and document analysis (Revision 2.0), Tech. Rep. TR-93-07, Information Science Research Institute, University of Nevada, Las Vegas, April 1993.

[18] R. Kasturi and L. O'Gorman, "Document image analysis: A bibliography", *Machine Vision and Appl.*, No. 5, 1992, pp. 231–243.

[19] W. Stallings, "Approaches to Chinese character recognition", *Pattern Recognition*, Vol. 8, 1976, pp. 87–98.

[20] S. Mori, K.Y. Yamamoto and N. Yasuda, "Research on machine recognition of handprinted characters", *IEEE Trans. Pattern Anal. Machine Intell.*, Vol. 6, No. 4, July 1984, pp. 386–405.

[21] A. Nazif, A system for the recognition of the printed Arabic characters, Master's Thesis, Faculty of Engineering, Cairo University, 1975.

[22] K. Badi and M. Shimura, "Machine recognition of Arabic cursive scripts", *Pattern Recognition Practice*, 1979, pp. 315–323.

[23] A. Nouh, A. Sultan and R. Tolba, "An approach for Arabic character recognition", *J. Engrg. Sci. King Saud Univ.*, Vol. 6, No. 2, 1980, pp. 185–191.

[24] A. Amin, A. Kaced, J. Haton and R. Mohr, "Hand written Arabic character recognition by the I.R.A.C. system", *Proc. 5th Internat. Conf. Pattern Recognition*, Miami, FL, 1980, pp. 729–731.

[25] A. Amin, "State of the art on character recognition", *Arabic Language Meeting*, Paris, IBM, Europe, 22–24 January 1985.

[26] A. Shoukry, "Arabic character recognition state of the art", *Proc. 11th National Computer Conf.*, Dhahran, Saudi Arabia, March 1989, pp. 382–390.

[27] K.M. Jambi, "Arabic character recognition: Many approaches and one decade", *Arabian J. Engrg. Sci.*, Vol. 16, No. 4, 1991, p. 499.

[28] I. Tekhonova, "Arabic optical character recognition", *Proc. 3rd Internat. Conf. and Exhibition on Multi-lingual Computing (Arabic and Roman Script)*, University of Durham, UK, December 1992, pp. 7.2.1–7.2.7 (Commercial System).

[29] P. Ahmed and M.A.A. Khan, "Computer recognition of Arabic scripts based text – The state of the art", *Proc. 4th Internat. Conf. and Exhibition on Multi-lingual Computing (Arabic and Roman Script)*, University of Cambridge, London, UK, April 1994, pp. 2.2.1–2.2.15.

[30] A. Emam, M. Ismail, H. AlKhatib and E. Korany, "Character recognition of Arabic scripts", *Proc. 4th Internat. Conf. and Exhibition on Multi-lingual Computing (Arabic and Roman Script)*, University of Cambridge, London, UK, April 1994, pp. 2.1.1–2.1.10.

[31] M.C. Fehri and M. Ben Ahmed, "A new approach to Arabic character recognition in multifont documents", *Proc. 4th Internat. Conf. and Exhibition on Multi-lingual Computing (Arabic and Roman Script)*, University of Cambridge, London, UK, April 1994, pp. 2.5.1–2.5.7.

[32] H. Goraine and M. Usher, "Printed Arabic text recognition", *Proc. 4th Internat. Conf. and Exhibition on Multilingual Computing (Arabic and Roman Script)*, University of Cambridge, London, UK, April 1994, pp. 2.6.1–2.6.8.

[33] K.M. Hassibi, "Machine-printed Arabic OCR using neural networks", *Proc. 4th Internat. Conf. and Exhibition on Multi-lingual Computing (Arabic and Roman Script)*, University of Cambridge, London, UK, April 1994, pp. 2.3.1–2.3.12.

[34] S.A. Ali, Topological analysis in the design of a machine to recognise handprinted characters, Ph.D. Thesis, Brunel University, England, 1979.

[35] A. Amin, "Arabic handwriting recognition and understanding", *Proc. Computer Processing and Transmission of the Arabic Language Workshop*, Kuwait, 14–16 April 1985, pp. 1–37.

[36] A. Amin, "IRAC: Recognition and understanding systems", in: R. Descout, ed., *Applied Arabic Linguistics and Signal and Information Processing*, Hemisphere, New York, 1987, pp. 159–170.

[37] T.S. El-Sheikh and S.G. El-Taweel, "Real-time Arabic handwritten character recognition", *Proc. 3rd Internat. Conf. on Image Processing and its Applications*, Warwick, UK, IEE. London, UK, July 1989, pp. 212–216.

[38] T.S. El-Sheikh and S.G. El-Taweel, "Real-time Arabic handwritten character recognition", *Pattern Recognition*, Vol. 23, No. 12, 1990, pp. 1323–1332.

[39] M.S. El-Wakil and A. Shoukry, "On-line recognition of handwritten isolated Arabic characters", *Proc. 1st King Saud University Symposium on Computer Arabization*, Riyadh, Saudi Arabia, April 1987, pp. 109–120.

[40] M.S. El-Wakil and A. Shoukry, "On-line recognition of handwritten isolated Arabic characters", *Pattern Recognition*, Vol. 22, No. 2, 1989, pp. 97–105.

[41] S. Saadallah and S. Yacu, "Design of an Arabic character reading machine", *Proc. Computer Processing and Transmission of the Arabic Language Workshop*, Kuwait, 14–16 April 1985.

[42] T.S. El-Sheikh, "Recognition of handwritten Arabic mathematical formulas", *Proc. UK IT 1990 Conf.*, Southampton, UK, March 1990, pp. 344–351.

[43] A. Amin, "Recognition of Arabic handprinted mathematical formulae", *Arabian J. Engrg. Sci.*, Vol. 16, No. 4, 1991, p. 531.

[44] K. Badi and M. Shimura, "Machine recognition of Arabic cursive scripts", *Trans. Institute Electronics Commun. Engrs. Japan*, Vol. E65, 1982, pp. 107–114.

[45] A. Amin, "Machine recognition of handwritten Arabic words by the IRAC II System", *Proc. 6th Internat. Joint Conf. on Pattern recognition*, Munich, FRG, October 1982, pp. 34–36.

[46] A. Amin and G. Masini, "Machine recognition of cursive Arabic words", in: *Application of Digital Image Processing IV*, San Diego, CA, August 1982, Vol. SPIE-359, pp. 286–292.

[47] S.I. Shaheen and G.A. Abo Samra, "On-line Arabic cursive script recognition", *Egyptian Comput. J.*, Vol. 18, No. 2, December 1990, pp. 222–240.

[48] S. Al-Emami and M. Usher, "On-line recognition of handwritten Arabic characters", *IEEE Trans. Pattern Anal. Machine Intell.*, Vol. 12, No. 7, July 1990, pp. 704–710.

[49] I.S. Abuhaiba and P. Ahmed, "Restoraion of temporal information in off-line Arabic handwriting", *Pattern Recognition*, Vol. 26, No. 7, July 1993, pp. 1009–1017.

[50] S.H. Abbas, M.H. Al Muifraje and M.I. Harba, "Optimising the digital learning network for recognition of the hand-written numerals used by Arabs", *Proc. European Conf.*, Paris, France, April 1986, pp. 505–513.

[51] H. Goraine, M. Usher and S. Al-Emami, "Off-line Arabic character recognition", *IEEE Comput.*, Vol. 25, No. 7, 1992, pp. 71–74.

[52] A. Amin and G. Masini, "Machine recognition of multi-font printed Arabic texts", *Proc. 8th Internat. Joint Conf. on Pattern Recognition*, Paris, France, October 1986, pp. 392–395.

[53] F. Haj-Hassan, "Arabic character recognition", in: P.A. MacKay, ed., *Computers and the Arabic Language*, Hemisphere, New York, 1985, pp. 113–118.

[54] A. Nouh, A. Nurul-Ula and A. Sharaf Eldin, "Boolean recognition technique for typewritten Arabic character set", *Proc. 1st King Saud University Symp. on Computer Arabization*, Riyadh, Saudi Arabia, April 1987, pp. 98–108.

[55] A. Nouh, A. Nurul-Ula and A. Sharaf Eldin, "A proposed algorithm for thinning binary Arabic character patterns", *Proc. 1st Kuwaiti Computer Conf.*, Kuwait, 27–29 March 1989, pp. 426–441.

[56] H.Y. Abdelazim and M.A. Hashish, "Automatic recognition of handwritten Hindi numerals", *Proc. Comp EURO '89 VLSI and Computer Peripherals*, Hamburg, West Germany, May 1989, pp. 287–298.

[57] H.Y. Abdelazim, A.M. Mousa, Y.R. Saleh and M.A. Hashish, "Arabic text recognition using a partial observation approach", *Proc. 12th National Computer Conf.*, Riyadh, Saudi Arabia, 21–24 October 1990, pp. 427–437.

[58] H.S. Al-Yousefi and S.S. Udpa, "Recognition of hand-written Arabic characters", *Proc. SPIE 32nd Annual Internat. Technical Symp. on Optical and Optoelectronic Applied Science and Engineering*, San Diego, CA, August 1988, Vol. 974, pp. 330–336.

[59] A. Amin and S. Al-Fedaghi, "Machine recognition of printed Arabic text utilizing natural language morphology", *IEEE Trans. Systems Man Cybernet.* Vol. 35, No. 6, 1991, pp. 769–788.

[60] K. Bouhilali, M. Kamrouni and N. Ellouze, "Method of segmentation of Arabic text image into characters", *Proc. 1st Kuwaiti Computer Conf.*, Kuwait, March 1989, pp. 442–446.

[61] S.S. El-Dabi, R. Ramsis and A. Kamel, "Arabic character recognition system: A statistical approach for recognizing cursive typewritten text", *Pattern Recognition*, Vol. 23, No. 5, 1990, pp. 485–495.

[62] T.S. El-Sheikh and R.M. Guindi, "Automatic recognition of isolated Arabic characters", *Signal Processing*, Vol. 14, No. 2, March 1988, pp. 177–184.

[63] I.S.I. Abuhaiba, Use of fuzzy set theory in pattern recognition with application to Arabic characters, Master's Thesis, University of Bradford, Bradford, England, 1990.

[64] H.S. Al-Yousefi and S.S. Udpa, "Recognition of hand-written Arabic characters via segmentation", *Arab Gulf J. Scient. Res.*, Vol. 8, No. 2, 1990, pp. 49–59.

[65] M.N. Al-Tikriti and S.K. Al-Ramahi, "A fuzzy approach for some Arabic handwritten characters computer recognition", *Proc. Computer Processing and Transmission of the Arabic Language Workshop*, Kuwait, 14–16 April 1985.

[66] A. El-Desouky, M. Salem, A. Abd El-Gwad and H. Arafat, "A handwritten Arabic character recognition technique for machine reader", *Internat. J. Mini Microcomput.*, Vol. 14, No. 2, 1992, pp. 57–61.

[67] S.S. Hyder and A. Koujah, "Character recognition of cursive scripts", *Proc. 1st Internat. Conf. on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems IEA/AIE - 88*, Tullahoma, TN, June 1988, pp. 1146–1150.

[68] H.Y. Abdelazim and M.A. Hashish, "Interactive font learning for Arabic OCR", *Proc. 1st Kuwaiti Computer Conf.*, Kuwait, March 1989, pp. 463–486.

[69] A. Goneid, M. Shalaby, S.K. Hindawi, T. Nazmi and K. Monir, "A fast algorithm for the recognition of Arabic handwritten numerals using topological features", *Proc. 1st Internat. Conf. on AI Applications*, Cairo, Egypt, 1992.

[70] K.M. Jambi, "A system for recognizing handwritten Arabic words", *Proc. 13th National Computer Conf.*, Riyadh, Saudi Arabia, November 1992, pp. 472–482.

[71] A. Zahour, B. Taconet and A. Faure, "A new method for recognition of Arabic cursive scripts", *Proc. 1st Internat. Conf. on Document Analysis and Pattern Recognition*, Saint-Malo, France, 1991, pp. 454–462.

[72] F. El-Khaly and M.A. Sid-Ahmed, "Machine recognition of optically captured machine printed Arabic text", *Pattern Recognition*, Vol. 23, No. 11, 1990, pp. 1207–1214.

[73] T.S. El-Sheikh and R.M. Guindi, "Computer recognition of Arabic cursive scripts", *Pattern Recognition*, Vol. 21, No. 4, 1988, pp. 293–302.

[74] H. Goraine and M. Usher, "Recognition of typewritten Arabic characters in different fonts", *Proc. IEE Colloquium on 'Character recognition and Applications'*, London, UK, October 1989, pp. 9/1–5.

[75] F. Khella, Analysis of hexagonally sampled images with application to Arabic cursive text recognition, Ph.D. Thesis, University of Bradford, Bradford, England, 1992.

[76] V. Margner, "SARAT – A system for the recognition of Arabic printed text", *Proc. 11th IAPR Internat. Conf. on Pattern Recognition*, The Hague, The Netherlands, September 1992, pp. 561–564.

[77] M.F. Tolba and E. Shaddad, "On the segmentation and recognition of printed Arabic characters", *Proc. 2nd Conf. on Arabic Computational Linguistic*, Kuwait, November 1989, pp. 490–507.

[78] M.F. Tolba and E. Shaddad, "On the automatic reading of printed Arabic characters", *Proc. IEEE Internat. Conf. on Systems Man Cybernet.*, Los Angeles, CA, 1990, pp. 496–498.

[79] S.H. El Ramly and M.A. El-Hamalaway, "A language dependent Arabic character recognition approach", *Proc. 14th Internat. Conf. on Statistics, Computer Science and Demographic Research*, Cairo, Egypt, March 1989, pp. 247–254.

[80] F. Haj-Hassan, "Printed Arabic text recognition", *Arabian J. Engrg. Sci.*, Vol. 16, No. 4, 1991, p. 551.

[81] H. Mahdi, "Thinning and transforming the segmented Arabic characters into meshes of unified small size", *Proc. 14th Internat. Conf. on Statistics, Computer Science and Demographic Research*, Cairo, Egypt, March 1989, pp. 263–276.

[82] S.A. Mahmoud, "Arabic character recognition using Fourier descriptors and character contour encoding", *Pattern Recognition*, Accepted.

[83] A. Nurul-Ula and A. Nouh, "Automatic recognition of Arabic characters using logic statements. Part I. System description and preprocessing", *J. Engrg. Sci. King Saud Univ.*, Vol. 14, No. 2, 1988, pp. 343–352.

[84] M.B. Fayek and B. Al-Basha, "A new hierarchical method for isolated typewritten Arabic character classification and recognition", *Proc. 13th National Computer Conf.*, Riyadh, Saudi Arabia, November 1992, pp. 750–760.

[85] H.Y. Abdelazim and M.A. Hashish, "Arabic typeset: An OCR approach", *Proc. 5th EUSIPCO-90, European Signal Processing Conf.*, Barcelona, Spain, September 1990, pp. 1019–1022.

[86] B. Al-Badr and R.M. Haralick, "Recognition without segmentation: Using mathematical morphology to recognize printed arabic", *Proc. 13th National Computer Conf.*, Riyadh, Saudi Arabia, November 1992, pp. 813–829.

[87] K. El Gowely, O. El Dessouki and A. Nazif, "Multi-phase recognition of multi-font photoscript Arabic text", *Proc. 10th Internat. Joint Conf. on Pattern Recognition*, 1990, pp. 700–702.

[88] B.M. Kurdy and A. Joukhadar, "Multifont recognition system for Arabic characters", *Proc. 3rd Internat. Conf. and Exhibition on Multi-lingual Computing (Arabic and Roman Script)*, University of Durham, UK, December 1992, pp. 7.3.1–7.3.9.

[89] M. Fakir and C. Sodeyama, "Machine recognition of Arabic printed scripts by dynamic programming matching method", *IEICE Trans. Inform. Systems*, Vol. 76, No. 2, February 1993, pp. 235–242.

[90] H.Y. Abdelazim and M.A. Hashish, "Automatic reading of bilingual typewritten text", *Proc. Comp EURO '89 VLSI and Computer Peripherals*, Hamburg, West Germany, May 1989, pp. 2/140–144.

[91] B. Parhami and M. Taraghi, "Automatic recognition of printed Farsi texts", *Pattern Recognition*, Vol. 14, No. 1, 1981, pp. 1–6.

[92] N. Yalabik and M. Ozcilingir, "Computer recognition of Ottoman text", *Computer and Information Sciences-3. Proc. ISCIS III. 3rd Internat. Symp. on Computer and Information Sciences*, Cesme, Turkey, November 1988, pp. 323–330.

[93] S.H. El Ramly and M.A. El-Hamalaway, "A new font for Arabic character simplifies recognition procedure", *Proc. 11th National Computer Conf.*, Dhahran, Saudi Arabia, March 1989, pp. 396–401.

[94] H.Y. Abdelazim and M.A. Hashish, "Arabic reading machine", *Proc. 10th National Computer Conf.*, King Abdulaziz University, Jiddah, Saudi Arabia, March 1988, pp. 733–743.

[95] F. Khella and S. Mahmoud, "Recognition of hexagonally sampled Arabic character", *Arabian J. Engrg. Sci.*, Accepted.

[96] A. Amin, "OCR of Arabic texts", *Proc. 4th Internat. Conf. on Pattern Recognition*, Cambridge, UK, March 1988, pp. 616–625.

[97] H.S. Baird, "Calibration of document image defect models", *Proc. 2nd Annual Symp. on Document Analysis and Information Retrieval*, Las Vegas, NV, April 1993, pp. 1–16.

[98] H.S. Baird, "Document image defect models", *Proc. Workshop on Syntactic and Structural Pattern Recognition 90*, Murrey Hill, NJ, June 1990, pp. 38–46.

[99] M.K. Brown and S. Ganapathy, "Preprocessing techniques for cursive script word recognition", *Pattern Recognition*, Vol. 16, No. 5, 1983, pp. 447–458.

[100] H.S. Baird, "Document image defect models and their uses", *Proc. Internat. Conf. on Document Analysis and recognition*, Nangano, Japan, October 1993, pp. 62–67.

[101] T. Kunango, R. Haralick and I. Phillips, "Global and local document degradation models", *Proc. Internat. Conf. on Document Analysis and Recognition*, Nangano, Japan, 1993.

[102] T. Pavlidis, "Effects of distortions on the recognition rate of a structural OCR system", *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, Washington, DC, June 1983, pp. 303–309.

[103] T. Pavlidis, "Recognition of printed text under realistic conditions", *Pattern Recog. Lett.*, Vol. 14, April 1993, pp. 37–326.

[104] R.M. Haralick and L.G. Shapiro, *Computer and Robot Vision*, Addison Wesley, Reading, MA, 1992, Vol. 1.

[105] V.P. Concepcion, M.P. Grzech and D.P. D'Amato, "Using morphology in document image processing", *Visual Communications and Image Processing '91: Image Processing*, Vol. SPIE-1606, 1991, pp. 132–140.

[106] T. Pavlidis, *Algorithms for Graphics and Image Processing*, Computer Science Press, Rockville, MD, 1982.

[107] S.A. Ali and M. Alsaadonn, "A parallel algorithm for thinning images", *Proc. 1st Kuwaiti Computer Conf.*, Kuwait, March 1989, pp. 121–140.

[108] C.J. Hilditch, "Linear skeletons from square cupboard", *Machine Intell.*, Vol. 4, 1969, pp. 403–420.

[109] S.A. Mahmoud, I. Abu Haiba and R.J. Green, "Skeletonization of Arabic characters using clustering based skeletonization algorithm (CBSA)", *Pattern Recognition*, Vol. 24, No. 5, 1991, pp. 453–464.

[110] N.J. Naccache and R. Shinghal, "SPTA: A proposed algorithm for thinning binary patterns", *IEEE Trans. Systems Man Cybernet.*, Vol. 14, No. 3, May 1984, pp. 409–418.

[111] M.Y. Jaisimha, R. Haralick and D. Dori, "A methodology for the characterization of the performance of thinning algorithms", *Proc. Internat. Conf. on Document Analysis and Recognition*, Nangano, Japan, 1993.

[112] L. Lam, S. Lee and C. Suen, "Thinning methodologies – A comprehensive survey", *IEEE Trans. Pattern Anal. Machine Intell.*, Vol. 14, No. 9, September 1992, pp. 869–885.

[113] W.H. Abdullah, A.O.M. Saleh and A.H. Morad, "A preprocessing algorithm for hand-written character recognition", *Pattern Recognition Lett.*, Vol. 7, January 1988, pp. 13–18.

[114] I.S. Abuhaiba, S.A. Mahmoud and R.J. Green, "Cluster number estimation and skeleton refining algorithms for Arabic characters", *Arabian J. Engrg. Sci.*, Vol. 16, No. 4, 1991, p. 519.

[115] S.H. El Ramly and M.A. El-Hamalaway, "A new font for Arabic character simplifies recognition procedure", *Proc. 14th Internat. Conf. Statistics, Computer Science and Demographic Research*, Cairo, Egypt, March 1989, pp. 255–261.

[116] T. Watanabe, L. Qin and N. Sugie, "Structure recognition methods for various types of documents", *Machine Vision Appl.*, Vol. 6, Nos. 2–3, 1993, pp. 163–176.

[117] H. Fujisawa, Y. Nakano and K. Kurino, "Segmentation methods for character recognition: From segmentation to document structure analysis", *Proc. IEEE*, Vol. 80, No. 7, July 1992, pp. 1079–1091.

[118] A. Amin, A. Kaced and J. Haton, "Arabic handwriting recognition by the IRAC system", *Comput. Arabization*, April 1988, pp. 9–11.

[119] M.B. Ahmad, A. Jaaly, G. Dreyfus and S. Knerr, "Recognizing Arabic characters using neural networks for electronic document processing", *Proc. Conf. on the Use of Arabic Language in Information Technology*, Riyadh, Saudi Arabia, May 1992 (in Arabic).

[120] A. Amin and J.F. Mari, "Machine recognition and correction of printed Arabic text", *IEEE Trans. Systems Man Cybernet.*, Vol. 19, No. 5, September 1989, pp. 1300–1306.

[121] C.E. Dunn and P.S.P. Wang, "Character segmentation techniques for handwritten text – A survey", *Proc. 11th IAPR Internat. Conf. on Pattern Recognition*, The Hague, The Netherlands, August 1992, Vol. 2, pp. 577–580.

[122] S.N. Srihari and R.M. Bozinvic, "A multi-level perception approach to reading cursive script", *Artificial Intell.*, No. 33, 1987, pp. 217–255.

[123] K.R. Tampi and S.S. Chetlur, "Segmentation of handwritten characters", *Proc. 8th Internat. Joint Conf. on Pattern Recognition*, Paris, France, October 1986, pp. 684–686.

[124] M. Maier, "Separating characters in scripted documents", *Proc. 8th Internat. Joint Conf. on Pattern Recognition*, Paris, France, October 1986, pp. 1056–1058.

[125] R.L. Hoffman and J.W. McCullough, "Segmentation methods for recognition of machine-printed characters", *IBM J. Res. Develop.*, Vol. 15, March 1971, pp. 153–165.

[126] R.G. Casey and G. Nagy, "Recursive segmentation and classification of composite character patterns", *Proc. 6th Internat. Joint Conf. on Pattern Recognition*, Munich, FRG, October 1982, pp. 1023–1026.

[127] D.G. Elliman and I.T. Lancaster, "A review of segmentation and contextual analysis techniques for text recognition", *Pattern Recognition*, Vol. 23, Nos. 2/3, 1990, pp. 337–346.

[128] H. Almuallim and S. Yamagochi, "A method of recognition of Arabic cursive handwriting", *Pattern Recognition*, Vol. 9, No. 5, September 1987, pp. 715–722.

[129] A. Amin, G. Masini and J.-P. Haton, "Recognition of handwritten Arabic words and sentences", *Proc. 7th Internat. Joint Conf. on Pattern Recognition*, October 1984, pp. 1055–1057.

[130] T.S. El-Sheikh and S.G. El-Taweel, "Segmentation of handwritten Arabic words", *Proc. 12th National Computer Conf.*, Riyadh, Saudi Arabia, 21–24 October 1990, pp. 389–402.

[131] A. Amin and H.B. Al-Sadoun, "A new segmentation technique of Arabic text", *Proc. 11th IAPR Internat. Conf. on Pattern Recognition*, The Hague, The Netherlands, September 1992, pp. 441–445.

[132] B. Al-Badr and R.M. Haralick, "Symbol recgonition withour prior segmentation", *Proc. IS&T/SPIE Symp. on Electronic Imaging Science and Techonogy Conf.: Document Recognition*, San Jose, CA, February 1994, Vol. 2181.

[133] H.Y. Abdelazim and A. Abdel-Mageed, "Automatic reading of Arabic text with spell checking assistance", *Proc. Conf. on the Use of Arabic Language in Information Technology*, Riyadh, Saudi Arabia, May 1992 (in Arabic).

[134] E.K. Alqaisy and H.L. Naser, "Recognition of Arabic numerals using probabilistic functions", *Proc. Computer Processing and Transmission of the Arabic Language Workshop*, Kuwait, 14–16 April 1985.

[135] A. Nouh, A. Nurul-Ula and A. Sharaf Eldin, "Algorithms for feature extraction: A case study for the Arabic character recognition", *Proc. 10th National Computer Conf.*, King Abdulaziz University, Jiddah, Saudi Arabia, March 1988, pp. 653–666.

[136] A. Nurul-Ula and A. Nouh, "Automatic recognition of Arabic characters using logic statements. Part II. Development of recognition algorithm", *J. Engrg. Sci. King Saud Univ.*, Vol. 14, No. 2, 1988, pp. 343–352.

[137] A. Nouh, A. Sultan and R. Tolba, "On feature extraction and selection for Arabic character recognition", *Arab Gulf J. Scient. Res.*, Vol. 2, No. 1, 1984, pp. 329–347.

[138] M.F. Tolba, S. Wahab and A. Salem, "A recognition algorithm for printed Arabic character", *Proc. IASTED Internat. Symp. in Applied Informatics*, Switzerland, 1987, pp. 128–131.

[139] R.O. Duda and P.E. Hart, *Pattern Classification and Scene analysis*, Wiley, New York, 1973.

[140] R. Schalkoff, *Pattern Recognition: Statistical, Structural and Neural Approaches*, Wiley, New York, 1992.

[141] R.P. Lippmann, "An introduction to computing with neural networks", *IEEE ASSP Mag.*, April 1987.

[142] S.Y. Kung, *Digital neural networks*, Prentice-Hall, Englewood Cliffs, NJ, 1993.

[143] D. Rumelhart, J. McClelland and the PDP Research Group, *Parallel Distributed Processing, Vols. 1 & 2*, MIT Press, Cambridge, MA, 1986.

[144] A.A. Nabawi and S.A. Mahmoud, "Recognition of Arabic characters using backpropagation neural network" (in Arabic), Submitted.

[145] H.E.-D. Mahgoub and H.Y. Abdelazim, "Arabic spelling verification and assistance", *Proc. Conf. on the Use of Arabic Language in Information Technology*, Riyadh, Saudi Arabia, May 1992.

[146] G. Nagy, "At the frontiers of OCR", *Proc. IEEE*, Vol. 80, No. 7, July 1992, pp. 1093–1100.

[147] I.T. Phillips, S. Chen and R.M. Haralick, "CD-ROM document database standard", *Proc. Internat. Conf. on Document Analysis and Recognition*, Nangano, Japan, October 1993, pp. 478–482.

[148] L.Xu, A. Krzyzak and C.Y. Suen, "Methods of combining multiple classifiers and their application to handwriting recognition", *IEEE Trans. Systems Man Cybernet.*, Vol. 22, No. 3, May/June 1992, pp. 418–435.

[149] H.Y. Abdelazim and M.A. Hashish, "Automatic recognition of Arabic text", *Proc. 10th Image/ITL Conf.*, IBM Toronto Lab, August 1987.

[150] H.Y. Abdelazim, Text recognition: Theory and implementation. Ph.D. Thesis, Cairo University, March 1989.

[151] A.O. Abd El-Gwad, M. Salem, F.A. Shadi and H. Arafat, "Automatic recognition of handwritten Arabic characters", *Proc. 25th Annual Conf. on Statistics, Computer Science and Operations Research*, Cairo University, Egypt, December 1990, pp. 4.63–4.75.

[152] S.S. Aiiah and S. Yaco, "Design of an Arabic character reading machine", *Comput. Arabization*, April 1988, pp. 53–69.

[153] B. Al-Badr, On the recognition of Arabic documents, Tech. Rep. 93-10-01, The Department of Computer Science and Engineering, University of Washington, Seattle, 1993.

[154] S. Al-Emami, Machine recognition of handwritten and typewritten Arabic characters, Ph.D. Thesis, Dept. of Cybernetics, University of Reading, September 1988.

[155] S. Al-Fedaghi and A. Amin, Automatic spelling correction in Arabic, Tech. rep., Electrical and Computer Engineering Department, Kuwait University, 1988.

[156] A.K. Al Jabri and A.S. Al Mohmoud, "Error-correction with unstructured redundancy: An application to written Arabic", *Proc. Conf. on the Use of Arabic Language in Information Technology*, Riyadh, Saudi Arabia, May 1992 (in Arabic).

[157] E.K. Alqaisy, Recognition of hand-written Arabic numerals using fast Fourier transform, Master's Thesis, National Center for Computers/Institute of Training and Research, Baghdad, Iraq, 1987 (in Arabic).

[158] M.N. Altikriti and V.S. Bansal, "Recognition of handwritten Arabic numerals using fuzzy entropy", *J. Engrg. Tech.*, Vol. 2, 1984, pp. 7–20.

[159] H.S. Al-Yousefi and S.S. Udpa, "Recognition of Arabic characters", *IEEE Trans. Pattern Anal. Machine Intell.*, Vol. 14, No. 8, August 1992, pp. 853–857.

[160] H.S. Al-Yousefi, Recognition of handwritten Arabic characters, Ph.D. Thesis, Colorado State University, Fort Collins, CO, Summer 1989.

[161] A. Amin, "Machine recognition of handwritten Arabic words by the IRAC II system", *Comput. Arabization*, April 1988, pp. 11–14.

[162] M.A. Assal and A.A. Fahmy, "Recognition of handwritten Hindu numerals in low quality images", *Proc. 1st Internat. Conf. on AI Applications*, Cairo, Egypt, 1992.

[163] M.M. Beglou, M.J. Holt and S. Datta, "Off-line cursive script recognition using a neural network", *Proc. 6th Internat. Conf. on Digital Processing of Signals in Communications*, Loughborough, England, September 1991, pp. 129–134.

[164] A. El-Desouky, M. Salem, A. Abd El-Gwad and H. Arafat, "A handwritten Arabic character recognition technique for machine reader", *Proc. 3rd Internat. Conf. on Software Engineering for Real Time Systems*, Cirencester, UK, September 1991, pp. 212–216.

[165] K. El Gowely, Recognition of multi-font photo script Arabic text, Master's Thesis, Faculty of Engineering, Cairo University, 1989.

[166] M.S. El-Wakil and A. Shoukry, "On-line recognition of handwritten isolated Arabic characters", *Comput. Arabization*, April 1988, pp. 70–82.

[167] A. Goneid, M.N. Mikhail, H. Kotkata and A.A. Hassan, "Optical character recognition of Arabic handwritten numerals", *Proc. 3rd Internat. Conf. and Exhibition on Multi-lingual Computing (Arabic and Roman Script)*, University of Durham, UK, December 1992, pp. 7.1.1–7.1.10.

[168] R. Guindi, An Arabic text recognition system, Master's Thesis, Cairo University, 1987.

[169] F. Haj-Hassan, "Arabic character recognition", *Comput. Arabization*, April 1988, pp. 15–20.

[170] S.S. Hyder and A. Koujah, "Character recognition of cursive scripts", *Proc. Regional Conf. on Informatics and Arabization (IRSIT)*, Tunis, June 1988, pp. 118–127.

[171] M.A. Hashish, A.T. Elkheshen and M.R. Elghonemy, "Experiences in isolated Arabic word recognition", *Proc. Computer Processing and Transmission of the Arabic Language Workshop*, Kuwait, April 1985, p. 80.

[172] K. Jambi and T. Grace, "A new topological structural approach for the recognition of an isolated Arabic word", *Proc. Information Technology in Support of Economic Development Conf.*, Khartoum, Sudan, December 1990.

[173] K.M. Jambi, Design and implementation of a system for recognizing Arabic handwritten words with learning ability. Ph.D. Thesis, Illinois Institute of Technology, Chicago, August 1991.

[174] M. Khemakhem and M. Fehri, "Arabic typewritten character recognition using dynamic comparison", *Proc. 1st Kuwaiti Computer Conf.*, Kuwait, March 1989, pp. 455–462.

[175] M. Khemakhem and M. Fehri, "Arabic typewritten character recognition using dynamic comparison", *Proc. 11th National Computer Conf.*, Dhahran, Saudi Arabia, March 1989, pp. 448–462.

[176] M.Y. Mahmoud, M.A. El-Hamalaway and A. Famy, "A statistical approach for Arabic character recognition", *Proc. 12th Internat. Conf. for Statistics and Computer Science*, Cairo, Egypt, 1987, Vol. 5, pp. 243–250.

[177] A. Murad, Arabic character recognition using microprocessor, Master's Thesis, University of Technology, Baghdad, Iraq, 1983.

[178] A. Nouh, A. Nurul-Ula and A. Sharaf Eldin, "Sequential processing of binary images", *Arabian J. Engrg. Sci.*, Vol. 13, No. 2, 1987, pp. 229–243.

[179] A. Nouh, A. Nurul-Ula and A. Sharaf Eldin, "The Arabic character database system, ACDABAS", *Proc. 1st King Saud University Symp. on Computer Arabization*, Riyadh, Saudi Arabia, April 1987, pp. 90–97.

[180] A. Nouh, A. Nurul-Ula and A. Sharaf Eldin, "An OCR based address reader for Arabic postal systems", *Proc. 1st Internat. Conf. of the Regional Institute for Informatics and Telecommunications*, Tunis, 9–11 March 1988.

[181] A. Nouh, A. Nurul-Ula and A. Sharaf Eldin, "Boolean recognition technique for typewritten Arabic character set", *Comput. Arabization*, April 1988, pp. 22–28.

[182] G. Rahho, S. Saadallah and S. Yac, "Bilingual hand-printed reading machine using a microcomputer", *Proc. 2nd Internat. Baghadad Conf. on Computer Technology and Applications*, Baghdad, Iraq, 1986.

[183] R. Ramsis, S. El-Dabi and A. Kamel, Arabic character recognition system, IBM Tech. Report, No. KSC027, January 1988.

[184] M. Sabri, "Recognition of printed Arabic character using the architectural correspondence technique", *Proc. 2nd Internat. Baghadad Conf. on Computer Technology and Applications*, Baghdad, Iraq, 1986.

[185] M.A. Sharkawy, M.F. Tolba and E. Shaddad, "Fourier descriptors for printed Arabic character recognition", *Proc. 13th Internat. Conf. for Statistics and Computer Science*, Cairo, Egypt, 1988, Vol. 5, pp. 137–146.

[186] A. Shoukry, "A sequential algorithm for the segmentation of typewritten Arabic digitized text", *Arabian J. Engrg. Sci.*, Vol. 16, No. 4, 1991, p. 543.

[187] M. Shridhar and A. Badreldin, "A high-accuracy syntatic recognition algorithm for handwritten numerals", *IEEE Trans. Systems Man Cybernet.*, Vol. 15, No. 1, January/February 1985, pp. 152–158.

[188] B. Tawfik, M. Kouta and M. Assal, "Recognition of handwritten Hindu numerals", *Proc. ORMA Conf. MTC*, Cairo, Egypt, 1991.

[189] M.F. Tolba, A. Goneid, A. Salem and E. Shaddad, "The sensitivity of some recognition algorithms for printed Arabic characters", *Proc. 13th IFIP Conf.*, Tokyo, Japan, 1987.

[190] M.F. Tolba, A. Goneid, A. Salem and E. Shaddad, "The sensitivity of some recognition algorithms for printed Arabic characters", *Proc. 12th Internat. Conf. for Statistics and Computer Science*, Cairo, Egypt, 1987, Vol. 5, pp. 231–241.

[191] M.F. Tolba, S. Wahab and A. Salem, "A recognition algorithm for printed Arabic characters", *Comput. Arabization*, April 1988, pp. 5–8.

[192] S.G. Yacy, Design and implementation of English reading machine for the blind and Arabic character recognition, Master's Thesis, University of Baghdad, Iraq, 1985.

[193] M.I. Yousef, Recognition of hand-written Indian numerals, Master's Thesis, Technology University, Baghdad, Iraq, 1986 (in Arabic).

[194] Y.M. Youssef, "An expert knowledge-based system for Arabic OCR", *Proc. 14th Internat. Conf. on Statistics, Computer Science and Demographic research*, Cairo, Egypt, March 1989, pp. 227–245.

[195] Y.M. Youssef, "Arabic optical character recognition (A.OCR)", *AlHandasah*, No. 12, February 1993, pp. 18–23 (in Arabic).

[196] A. Zahour, B. Taconet and A. Faure, "Syntactic method for recognition of Arabic cursive writing", *Proc. IASTED Internat. Symp. Applied Informatics – AI '89*, Grindlewald, Switzerland, February 1989, pp. 97–101 (in French).

[197] F.W. Zaki, S.H. El-Konyaly, A.I. Abd El-Fattah and Y. Enab, "A new technique for Arabic handwriting recognition", *Proc. 11th Internat. Conf. for Statistics and Computer Science*, Cairo, Egypt, 1986, Vol. 2, pp. 171–180.